UNIVERSIDAD COMPLUTENSE DE MADRID

FACULTAD DE INFORMÁTICA
Departamento de Arquitectura de Computadoras y Automática

DESARROLLO DE NUEVAS TECNOLOGÍAS PARA EL AJUSTE DE ESTRUCTURAS TRIDIMENSIONALES EN BIOMOLÉCULAS SOBRE INFRAESTRUCTURAS GRID.

MEMORIA PARA OPTAR AL GRADO DE DOCTOR

PRESENTADA POR

José Ignacio Garzón Cañas

Bajo la dirección de los doctores

Pablo Chacón
Rubén Santiago Montero

Madrid, 2010

ISBN: 978-84-693-6333-1

Universidad Complutense de Madrid

Facultad de Informática

Departamento de Arquitectura de Computadores y Automática

Desarrollo de nuevas metodologías para el ajuste de estructuras tridimensionales en biomoléculas sobre infraestructuras Grid

Tesis Doctoral

José Ignacio Garzón Cañas

2009

Universidad Complutense de Madrid

Facultad de Informática

Departamento de Arquitectura de Computadores y Automática

Memoria presentada para optar al grado de Doctor

por

José Ignacio Garzón Cañas

Esta tesis doctoral ha sido financiada por el programa de Contratos de Personal Investigador de la Comunidad de Madrid. Su realización se ha llevado a cabo en el Centro de Investigaciones Biológicas (CSIC) en colaboración con el Departamento de Arquitectura de Computadores y Automática de la Universidad Complutense de Madrid bajo la dirección de los Drs. Pablo Chacón y Rubén Santiago Montero.

Firma del Doctorando	**Conformidad del Co-director**	**Conformidad del Co-director**
José Ignacio Garzón	**Dr. Pablo Chacón**	**Dr. Rubén Santiago Montero**

A mis padres.

All of old. Nothing else ever. Ever tried.
Ever failed. No matter. Try again.
Fail again. Fail better.
Samuel Beckett. Worstward Ho

Agradecimientos

Cualquier agradecimiento que pueda dar tiene que comenzar forzosamente por mis padres. A ellos debo la educación, el cariño y el apoyo incondicional que me han permitido llegar hasta aquí. También hago extensible este agradecimiento a mis hermanos y a mis tíos y primos.

No podría haber llevado a cabo este trabajo sin la ayuda de varios colaboradores. Gran parte del desarrollo matemático de las aplicaciones de ajuste ha sido realizado gracias a los conocimientos de Julio Kovacs, que colaboró con nosotros cuando trabajaba en el laboratorio de Ruben Abagyan en el *Scripps Institute*. También tengo que agradecer la ayuda prestada por Juan Fernández Recio y Carles Pons del *Barcelona Supercomputing Center*, que nos permitió afrontar el estudio del potencial de desolvatación en el ajuste proteína-proteína. En cuanto a la parte de computación Grid, la disponibilidad de mi co-director Ruben Santiago Montero me ha facilitado en gran medida el trabajo, así como la asistencia técnica, siempre útil y rápida, de José Luis Vazquéz-Poletti, al que no sé cuántos mensajes habré podido enviar estos años pidiéndole ayuda. También quiero recordar a Ángel Ramírez Ortiz y a los miembros de su grupo. Si bien nuestra colaboración no ha podido ser, por tristes motivos, muy productiva, tengo que agradecerles su disponibilidad. Sirvan también estas líneas como un pequeño tributo a la memoria de Ángel. Finalmente, tengo que agradecer la colaboración de mis compañeros de grupo, Mon y Vanessa, y la dedicación, ayuda e infinita paciencia de mi jefe y co-director Pablo. A pesar de las discusiones y de que pueda parecer que casi nunca te hago caso, ha sido un verdadero honor trabajar contigo.

Además, no puedo dejar de mencionar a todos los compañeros del CIB que han hecho, con su apoyo y sentido del humor, que la tarea de realizar esta tesis haya sido un poco más llevadera. A algunos, como María, Elizabeth, Ernesto e Israel, les debo útiles consejos en la realización y edición de esta tesis. También quiero expresar mi agradecimiento a Rubén, Marian, Sonia, Marta, un par de Lauras, David, Benet, Ruth, Chiara, Sara, Felicia, Erney, Javier, Oscar, Bego, Ángel, Eva, Magi, Rafa, Carmen, Andrés, Cesar, Loli, Gisela, José Manuel, Isabel, Claudia, Antonio Javier y Fernando. Incluso estoy agradecido a Luque, aunque no sé muy bien por qué. Y para finalizar con los agradecimientos, también quiero recordar a los amigos, a los que están cerca y a los que están lejos, por su amistad y esas cervezas compartidas que siempre ayudan en los momentos de crisis. En especial tengo que citar a Alberto, que tiene una pequeña parte de culpa en el hecho de que me metiera en este lío y también ha colaborado en la revisión de esta tesis.

Por último, no puedo terminar estos agradecimientos sin incluir también mis disculpas a Pablo, Mon, Vanessa, Isabel, Chiara, Sara y, últimamente, Erney. Sé que trabajar en un laboratorio tan pequeño con alguien con un carácter tan "apacible" como el mío puede llegar a ser algo irritante. Afortunadamente, sois gente estupenda y habéis aprendido a tomároslo con sentido del humor. Ese es el secreto para tratar conmigo, no tomarme en serio jamás.

Índice

1 INTRODUCCIÓN 1

1.1 MOTIVACIÓN 1

1.2 OBJETIVOS 5

1.3 ESTRUCTURA 6

2 INTRODUCCIÓN A COMPUTACIÓN GRID Y APLICACIONES BIOINFORMÁTICAS 9

2.1 COMPUTACIÓN GRID 9

2.1.1 MOTIVACIÓN Y EVOLUCIÓN DE LA COMPUTACIÓN GRID 10

2.1.2 CARACTERÍSTICAS DE LA COMPUTACIÓN GRID 12

2.1.3 GLOBUS TOOLKIT 15

2.1.4 METAPLANIFICACIÓN 17

2.1.4.1 Ejemplos de Metaplanificadores 20

2.1.4.2 GridWay 22

2.1.4.2.1 Arquitectura 23

2.1.4.2.2 Acciones de planificación 24

2.1.4.2.3 Ventajas de GridWay 27

2.2 BIOINFORMÁTICA Y COMPUTACIÓN GRID. 28

2.2.1 MÉTODOS DE AJUSTE TRIDIMENSIONAL Y APLICACIONES BIOINFORMÁTICAS. 29

2.2.1.1 Tipos de ajuste 30

2.2.1.2 Criterios de ajuste 31

2.2.1.3 Exploración 32

2.2.1.4 Aplicaciones bioinformáticas 33

3 SISTEMA CACHE DE "EXTREMO A EXTREMO" SOBRE METAPLANIFICACIÓN 37

3.1 APROXIMACIONES A FUNCIONALIDAD CACHE SOBRE SISTEMAS GRID 38

3.1.1 APROXIMACIONES SOFTWARE 38

3.1.2 APROXIMACIONES HARDWARE 39

3.1.3 FUNCIONALIDADES SIMILARES 40

3.2 CARACTERÍSTICAS DEL SISTEMA CACHE. 41

3.3 DISEÑO DEL SISTEMA CACHE 44
3.3.1 POLÍTICAS CACHE 46
3.3.1.1 Política de reemplazo 46
3.3.1.2 Políticas de transferencia 46
3.3.1.2.1 Política centralizada 48
3.3.1.2.2 Política remota 53
3.3.1.2.3 Comparación entre políticas 54
3.4 ADAPTACIÓN DEL SISTEMA CACHE SOBRE EL METAPLANIFICADOR GRIDWAY 55
3.4.1 POLÍTICA CENTRALIZADA EN GRIDWAY 56
3.4.2 POLÍTICA REMOTA EN GRIDWAY 59

4 APLICACIONES BIOINFORMÁTICAS DE AJUSTE 61

4.1 ESTRATEGIA DE EXPLORACIÓN. AJUSTE ROTACIONAL RÁPIDO. 61
4.1.1 AJUSTE ROTACIONAL RÁPIDO SOBRE TRES GRADOS DE LIBERTAD (FRM3) 63
4.1.2 AJUSTE ROTACIONAL RÁPIDO SOBRE CINCO GRADOS DE LIBERTAD (FRM5) 69
4.2 DESARROLLO DE APLICACIONES BIOINFORMÁTICAS 72
4.2.1 ADAPTACIÓN DE FRM3 AL AJUSTE MULTI-RESOLUCIÓN: ADP_EM 72
4.2.1.1 Función de Correlación 74
4.2.1.2 Origen de las expansiones armónicas 76
4.2.1.3 Delimitación del espacio traslacional 77
4.2.1.4 Esquema general de funcionamiento de ADP_EM 78
4.2.2 ADAPTACIÓN DE FRM3 AL AJUSTE PROTEÍNA-PROTEÍNA: FRODOCK 78
4.2.2.1 Función de Correlación 80
4.2.2.2 Origen de las expansiones armónicas 85
4.2.2.3 Delimitación del espacio traslacional 86
4.2.2.4 Esquema general de funcionamiento de FRODOCK 87
4.3 ADAPTACIÓN A COMPUTACIÓN GRID DE APLICACIONES BIOINFORMÁTICAS 87
4.3.1 APLICACIÓN ADP_EM 89
4.3.2 APLICACIÓN FRODOCK 92
4.3.3 ASPECTOS DE LA PARALELIZACIÓN EN COMPUTACIÓN GRID 95

5 VALIDACIÓN DE LAS APLICACIONES DE AJUSTE 97

5.1 VALIDACIÓN ADP_EM 98
5.1.1 VALIDACIÓN SOBRE MAPAS SIMULADOS 99
5.1.2 VALIDACIÓN SOBRE MAPAS EXPERIMENTALES 103
5.1.3 AJUSTE 6D CON MODELOS POR HOMOLOGÍA 104

5.2 VALIDACIÓN FRODOCK 107

6 EFICIENCIA DEL SISTEMA CACHE 115

6.1 CARACTERIZACIÓN DE EFICIENCIA 116
6.2 ENTORNO GRID EMPLEADO 119
6.3 ESTUDIO DE EFICIENCIA PARA FRODOCK 120
6.3.1 DESCRIPCIÓN DEL PROBLEMA EMPLEADO 120
6.3.2 ANÁLISIS DE LA EFICIENCIA EN TRANSFERENCIA 122
6.3.3 ANÁLISIS DE LA MEJORA DE PRODUCTIVIDAD DE LA APLICACIÓN 127
6.3.4 ANÁLISIS EN UN PROBLEMA DE MENOR COSTE 132
6.4 ESTUDIO DE EFICIENCIA PARA ADP_EM 134
6.4.1 DESCRIPCIÓN DEL PROBLEMA EMPLEADO 135
6.4.2 ANÁLISIS DE LA MEJORA DE PRODUCTIVIDAD 136
6.5 ESTUDIO DE EFICIENCIA SOBRE BENCHMARK GRID 139

7 CONCLUSIONES 145

7.1 PRINCIPALES APORTACIONES 145
7.2 PUBLICACIONES RELACIONADAS 148
7.3 TRABAJO FUTURO 149

APÉNDICE 151

BIBLIOGRAFÍA 157

Índice de figuras

Fig. 2.1- Esquema de la arquitectura Grid comparada con la arquitectura Internet..... 14
Fig. 2.2- Uso de un metaplanificador en arquitectura Grid..... 15
Fig. 2.3- Arquitectura de especificaciones seguida por Globus 15
Fig. 2.4- Fases en la planificación Grid para la ejecución de un trabajo..... 18
Fig. 2.5- Arquitectura CSF 20
Fig. 2.6- Arquitectura GSB..... 21
Fig. 2.7- Arquitectura GRMS 21
Fig. 2.8- Arquitectura VMS..... 22
Fig. 2.9- Arquitectura GridWay..... 24
Fig. 2.10- Esquema de detección y selección de recursos en GridWay 25
Fig. 2.11- Tipos de ajuste..... 30
Fig. 2.12- Esquema de búsqueda 6-Dimensional directa..... 32
Fig. 2.13- Ajuste multi-resolución interno..... 34
Fig. 2.14- Ajuste proteína-proteína externo 35

Fig. 3.1- Esquema del sistema cache de "extremo a extremo"..... 42
Fig. 3.2- Diferentes escenarios de almacenamiento en recursos Grid..... 47
Fig. 3.3- Base de datos del sistema cache mantenida por el metaplanificador..... 48
Fig. 3.4- Establecimiento de una tarea en un recurso con la política centralizada del sistema cache. 49
Fig. 3.5- Acciones realizadas por la rutina de control para gestionar el repositorio en la política centralizada del sistema cache. 50
Fig. 3.6- Acciones del metaplanificador tras la ejecución correcta de la tarea con la política centralizada del sistema cache. 51
Fig. 3.7- Actualización de base de datos por parte del metaplanificador en la política centralizada del sistema cache 52
Fig. 3.8- Esquema de ejecución de una tarea con la política remota del sistema cache 53
Fig. 3.9- Concurrencia en la transferencia de un fichero a un mismo recurso por parte de dos tareas..... 58

Fig. 4.1- Esquema de búsqueda 6-Dimensional con aceleración en el espacio traslacional (FTM)..... 62
Fig. 4.2- Esquema de búsqueda 6-Dimensional acelerando el espacio rotacional (FRM) 63
Fig. 4.3- Muestreo esférico y radial de una propiedad de un objeto tridimensional 68
Fig. 4.4- Esquema de realización de un proceso de Ajuste Rotacional Rápido (ADP_EM) 69
Fig. 4.5- Exploración del espacio 6-Dimensional mediante dos rotaciones y una traslación..... 70
Fig. 4.6- Modelado de la estructura atómica de un complejo macromolecular mediante ajuste multi**-**resolución. 73
Fig. 4.7- Combinación de ajuste multi-resolución con herramientas de modelado por homología 74
Fig. 4.8- Ilustración del cálculo de correlación empleado en el ajuste multi-resolución..... 75
Fig. 4.9- Esquema ilustrativo de la determinación del origen para las expansiones armónicas radiales en cada punto traslacional en el ajuste multi-resolución..... 76
Fig. 4.10- Delimitación del espacio traslacional en el ajuste externo. 77
Fig. 4.11- Esquema de funcionamiento de la aplicación ADP_EM 79
Fig. 4.12- Potencial de van der Waals 82

Fig. 4.13- Potencial Electroestático. 83
Fig. 4.14- Potencial de desolvatación. 84
Fig. 4.15- Establecimiento del origen de las expansiones armónicas radiales sobre el receptor y sobre el ligando en ajuste externo 85
Fig.4.16- Esquema de determinación de posiciones traslacionales válidas entre receptor y ligando en ajuste externo..... 86
Fig.4.17- Esquema de funcionamiento de la aplicación FRODOCK. 88
Fig.4.18- Realización de una evaluación de modelos por homología mediante ADP_EM en un entorno Grid. 90
Fig. 4.19- Partición del espacio de búsqueda traslacional en la paralelización de FRODOCK. 92
Fig. 4.20- Realización de un ajuste proteína-proteína mediante FRODOCK en un entorno Grid. 94

Fig. 5.1- Ejemplos de ajuste multi-resolución con datos simulados a partir de la estructura conocida del complejo. 98
Fig. 5.2- Precisión media del ajuste en el conjunto de casos simulados..... 100
Fig. 5.3- Resultados de ajuste multi-resolución sobre mapas a baja resolución obtenidos experimentalmente por EM. 102
Fig. 5.4- Esquema de validación del ajuste multi-resolución de modelos homólogos 105
Fig. 5.5- Relación entre correlación, obtenida por ADP_EM, y alineamiento estructural, obtenido por MAMMOTH, en el caso 1MUP 106
Fig. 5.6- Evaluación de un ajuste proteína-proteína mediante $RMSD_L$ y $RMSD_I$..... 108
Fig. 5.7- Porcentaje de acierto obtenido, evaluando mediante $RMSD_L$, en el conjunto de pruebas empleado para evaluar FRODOCK. 109
Fig. 5.8 Porcentaje de acierto obtenido, evaluando mediante $RMSD_I$, en el conjunto de pruebas empleado para evaluar FRODOCK. 110
Fig. 5.9- Comparación entre los resultados obtenidos en los ejemplos de validación CAPRI mediante FRODOCK y las estructuras originales del complejo..... 112

Fig. 6.1- Mejor ajuste encontrado empleando FRODOCK en el caso de prueba computacionalmente costoso 1NC2 120
Fig. 6.2- Eficiencia media obtenida sobre diferentes recursos en función del número de tareas en los que ha sido dividido el problema 1N2C 124
Fig. 6.3- Tiempo operacional total necesario para realizar todas las tareas en función del número de tareas en los que ha sido dividido el problema 1NC2. 126
Fig. 6.4- Media y desviación estándar de los tiempos de ejecución en función del número de tareas en los que el problema ha sido dividido (NT) para el caso 1N2C..... 127
Fig. 6.5- Ajuste lineal del comportamiento medio del sistema Grid para 1N2C (NT=200) con y sin cache... 128
Fig. 6.6- Comparación entre rendimiento teórico y experimental medio para ejecuciones de FRODOCK con 1N2C (NT=200) realizadas sobre el sistema Grid con y sin cache 128
Fig. 6.7- Tiempos medios empleados, en cada recurso para transferir los ficheros de entrada y ejecutar cada tarea de FRODOCK (caso 1N2C, NT=200) con y sin cache 129
Fig. 6.8- Número de tareas realizadas en cada recurso entre los cinco experimentos con y sin cache (caso 1N2C, NT=200) 130
Fig. 6.9- Eficiencias en trasferencia obtenidas en los recursos (caso 1N2C, NT=200)..... 130
Fig. 6.10- Proporción de acierto en búsqueda de ficheros en cache para cada recurso (caso 1N2C, NT=200) 131
Fig. 6.11- Media y desviación estándar de los tiempos de ejecución en función del número de tareas en los que el problema ha sido dividido (NT) para el caso 1PPE..... 132

Fig. 6.12- Ajuste lineal del comportamiento (tareas realizadas por segundo) medio del sistema Grid para 1PPE (NT=150) con y sin cache. 133
Fig. 6.13- Comparación entre rendimiento teórico y experimental medio para ejecuciones de FRODOCK con 1PPE (NT=150) realizadas sobre el sistema Grid con y sin cache. 133
Fig. 6.14- Resultados de ajuste ADP_EM para evaluación de modelos de homología para el caso 2CMD..... 136
Fig. 6.15- Ajuste lineal del comportamiento (tareas realizadas por segundo) medio del sistema Grid para 2CMD (NT=300) con y sin cache. 137
Fig. 6.16- Comparación entre rendimiento teórico y experimental medio para ejecuciones de FRODOCK con 2CMD (NT=300) realizadas sobre el sistema Grid con y sin cache. 137
Fig. 6.17- Tiempos medios empleados, para cada recurso del sistema, en transferir los ficheros de entrada y ejecutar cada tarea para ejecuciones de ADP_EM con y sin cache (2CMD. NT=300). 138
Fig. 6.18- Número de tareas ejecutadas en cada recurso entre los 5 experimentos ADP_EM con 300 modelos con y sin cache (2CMD). 138
Fig. 6.19- Acierto medio en la búsqueda de datos de entrada en cache (ADP_EM con 300 modelos. 2CMD)..... 138
Fig. 6.20- Eficiencias en trasferencia obtenidas en los recursos (ADP_EM con 300 modelos. 2CMD). 139
Fig. 6.21- Esquema de simulación de aplicación de alta productividad con reutilización de datos mediante problema ED. 140
Fig. 6.22- Rendimientos medios obtenidos con y sin cache para las distintas clases del problema ED. 141
Fig. 6.23- Tiempo operacional total para cada clase del problema ED y en cada recurso con y sin cache. .. 142

Índice de tablas

Tabla 2.1- Tipos de problema con alto requerimiento computacional.....10
Tabla 2.2- Modelos de computación en red. 10
Tabla 2.3- Componentes de Globus Toolkit versión 4 16

Tabla 5.1- Tiempos medios, en segundos, obtenidos para ajuste multi-resolución del conjunto de casos simulados. 102
Tabla 5.2- Tiempos medios empleados en el ajuste multi-resolución sobre mapas experimentales con ADP_EM y 3SOM.....104
Tabla 5.3- Resultados obtenidos con FRODOCK en el ajuste de casos CAPRI 111

Tabla 6.1- Características de recursos Grid empleados. 119
Tabla 6.2- Datos experimentales obtenidos para el problema 1N2C en diferentes recursos y con diferentes valores de NT. 123
Tabla 6.3- Rendimientos obtenidos para diferente número de tareas (NT) 129
Tabla 6.4- Tiempos de ejecución para la aplicación ADP_EM en entorno Grid con y sin sistema cache. .136
Tabla 6.5- Dimensiones de las matrices tridimensionales y tamaño de los ficheros para cada clase de problemas ED. 141

Tabla A.1- Resultados obtenidos con FRODOCK sobre el conjunto de pruebas estándar empleando como criterio de evaluación el cálculo de $RMSD_L$. 152
Tabla A.2- Resultados obtenidos con FRODOCK sobre el conjunto de pruebas estándar empleando como criterio de evaluación el cálculo de $RMSD_I$ 154

Acrónimos

ADP_EM: Another Docking Platform- Electron Microcopy.

CAPRI: Critical Assessment of PRediction of Interactions.

COLORES: COrrelation based LOw RESolution.

EGEE: Enabling Grids for E-sciencE.

EM: Electron Microscopy.

FFT: Fast Fourier Transform.

FRM: Fast Rotational Matching.

FRODOCK: Fast ROtational DOCKing.

FTM: Fast Translational Matching.

GASS: Global Access to Secondary Storage.

GRAM: Grid Resource Allocation and Management.

GridFTP: Grid File Transfer Protocol.

GT: Globus Toolkit.

HTTP: HyperText Transfer Protocol.

MD5: Message-Digest algorithm 5.

MDS: Monitoring and Discovery Service.

NT: Número de Tareas paralelas en las que un problema es dividido.

NP: Número de Posiciones por tarea en una paralelización de FRODOCK.

PDB: Protein Data Bank.

PQS: Protein Quaternary Structure file server.

RFT: Reliable File Transfer.

RMSD: Root Mean Square Deviation.

SASA: Solvent Accessible Surface Area.

SO(3): Tridimensional rotation Special Orthogonal group.

VO: Virtual Organization.

1 Introducción

La presente tesis doctoral resume el trabajo desarrollado en el Grupo de Bioinformática Estructural del Centro de Investigaciones Biológicas (CSIC) en colaboración con el Departamento de Arquitectura de Computadores y Automática de la Facultad de Informática de la Universidad Complutense de Madrid. Este trabajo ha tenido como principal objetivo la mejora de la productividad de sistemas Grid mediante la incorporación de un sistema cache de fácil implantación. Este sistema cache ha sido diseñado para aprovechar las funcionalidades proporcionadas por metaplanificadores de tareas, de forma que su inclusión sea simple y no requiera importantes modificaciones de los servicios Grid existentes. La mejora proporcionada será obtenida principalmente en aplicaciones paralelizadas de alta productividad en las que parte de la información de entrada sea repetidamente empleada. En este sentido, un segundo objetivo es el desarrollo y adaptación de aplicaciones bioinformáticas de ajuste tridimensional para su realización sobre computación Grid. Gracias al uso del sistema cache, se obtendrá un marco eficiente que permitirá resolver problemas de gran interés en biología estructural.

1.1 Motivación

El gran crecimiento de las redes de comunicación en las últimas décadas junto a los avances tecnológicos experimentados, ha proporcionado nuevos paradigmas para afrontar la realización de aplicaciones con alto coste computacional. Así, la computación Grid se ha establecido como alternativa a la utilización de grandes recursos locales para la obtención de alta capacidad computacional. La computación Grid posibilita la compartición de recursos entre distintas administraciones de forma segura y estable, facilitando el acceso a capacidades computaciones elevadas y heterogéneas sin la necesidad de asumir los costes derivados de su adquisición y mantenimiento. Esto es especialmente ventajoso cuando las necesidades computacionales son puntuales. En tales casos, el

mantenimiento de grandes infraestructuras locales puede implicar un desaprovechamiento de las mismas. Además, la capacidad de crecimiento de un sistema Grid no puede ser proporcionada por soluciones de índole local.

Existe una gran variedad de aplicaciones en diferentes áreas cuya adaptación a sistemas Grid supone un importante incremento de su productividad o de su capacidad de respuesta. Aplicaciones de respuesta en tiempo real, de tratamiento de grandes cantidades de información o de cálculo intensivo necesitan amplias capacidades computacionales para proporcionar soluciones en un tiempo de respuesta adecuado. Entre ellas y dentro del área de la bioinformática, se encuentran aplicaciones de búsqueda y ajuste tridimensional orientadas a la modelización y predicción de estructuras atómicas de moléculas y complejos macromoleculares.

La presente tesis doctoral está orientada a resolver eficazmente este tipo de problemas bioinformáticos aprovechando las capacidades proporcionadas por la computación Grid. Para ello se han desarrollado dos nuevas aplicaciones que resuelven de forma eficiente dos tipos de ajuste:

- ***Ajuste interno: Modelización de estructura de complejos macromoleculares.*** Realiza el ajuste interno de estructuras atómicas sobre mapas a baja resolución obtenidos por Microscopía Electrónica. Este tipo de ajuste proporciona un modelo de la estructura atómica de complejos macromoleculares a partir de sus componentes. La aplicación desarrollada para este tipo de ajuste es llamada ADP_EM (*Another Docking Program-Electron Microscopy*).
- ***Ajuste externo: Predicción de interacciones entre proteínas mediante ajuste externo.*** Realiza el ajuste externo entre dos estructuras atómicas con el fin de determinar la conformación del complejo basándose en la minimización de la energía libre del sistema. La aplicación que realiza este ajuste es llamada FRODOCK (*Fast ROtational DOCKing*).

Ambas aplicaciones han sido diseñadas desarrollando una novedosa metodología basada en el uso de esféricos armónicos que permite la aceleración de la búsqueda del mejor ajuste en el espacio rotacional. Estos métodos proporcionan resultados correctos en tiempos que, comparados con aplicaciones precedentes, suponen una importante mejora. Sin embargo, aún así la resolución de estos problemas puede requerir tiempos de cómputo elevados. Además, es

frecuente la necesidad de múltiples ajustes variando parte de los datos de entrada. De este modo, la aplicación de estas herramientas puede necesitar computaciones de días o incluso semanas. Por ello, las aplicaciones han sido rediseñadas para permitir su ejecución paralela sobre computación Grid. Básicamente, la paralelización permitirá la exploración concurrente de diferentes subconjuntos del espacio de búsqueda del ajuste. Teniendo en cuenta las características de las aplicaciones, las adaptaciones paralelas realizadas han sido:

- ***ADP_EM***: Si bien la realización de un ajuste interno mediante ADP_EM no supone un tiempo de cómputo excesivo, en casos prácticos puede ser necesario realizar el ajuste de miles de estructuras diferentes sobre un mismo mapa a baja resolución. ADP_EM ha sido adaptado para permitir el ajuste paralelo de múltiples estructuras empleando la misma información preprocesada del mapa a baja resolución.
- ***FRODOCK***: El espacio traslacional que define la posición de una proteína con respecto a la otra es dividido, de forma que distintas tareas exploran diferentes regiones del espacio traslacional.

Las aplicaciones, así adaptadas, pueden considerarse aplicaciones de alta productividad ya que su resolución se caracteriza por la realización reiterada de un mismo proceso de cálculo sobre diferente información de entrada (distintas estructuras a ajustar, distintas posiciones). Sin embargo, en ambos casos parte de la información empleada permanece invariable en todos los cálculos realizados. En el caso de ADP_EM el mapa a baja resolución siempre es el mismo. En FRODOCK las mismas estructuras atómicas son empleadas en todas las tareas paralelas. Este reiterado uso de la misma información puede producir la transferencia repetitiva de datos a un mismo recurso si varias tareas le son asignadas para su ejecución. Dada la naturaleza distribuida de los sistemas Grid, donde los recursos pueden encontrarse separados geográficamente por grandes distancias, esta transferencia repetitiva resulta ineficiente y afecta de manera significativa a la productividad del sistema.

Para mejorar la productividad de este tipo de aplicaciones se debe proporcionar al sistema Grid la capacidad de aprovechar, al realizarse la ejecución de una tarea en un recurso remoto, la información transmitida para otra tarea. Esta funcionalidad se ha obtenido mediante un novedoso sistema cache que permite mantener repositorios de datos en todos los recursos empleados del sistema Grid.

De esta forma, cuando una tarea se ejecuta en un recurso, se determina previamente si la información que necesita está accesible en el repositorio, evitándose la transferencia en caso de que así sea. Además, el sistema ha sido diseñado para aprovechar la funcionalidad ofrecida por los sistemas de metaplanificación. Un metaplanificador, de forma simplificada, interactúa con las funcionalidades a distintos niveles de un sistema Grid para proporcionar a los usuarios un marco de ejecución adecuado, facilitando el uso de dicho sistema. La integración del sistema cache en un metaplanificador permite proporcionar las funcionalidades necesarias para la gestión de los repositorios remotos sin la necesidad de realizar modificaciones intrusivas, costosas y difíciles de aplicar sobre el *software* y el *hardware* que integran un sistema Grid.

Mediante este sistema cache, adecuadamente integrado en el metaplanificador GridWay, se han realizado pruebas empleando las aplicaciones bioinformáticas anteriormente indicadas. Estos experimentos han permitido determinar que el uso del sistema cache proporciona reducciones significativas del tiempo de ejecución. Igualmente, se ha determinado mediante un conjunto de pruebas estándar para computación Grid la conveniencia de este sistema en aplicaciones generales con alta reusabilidad de datos.

1.2 Objetivos

El principal objetivo de la tesis doctoral es el diseño e implementación de un nuevo sistema cache aplicado a la tecnología Grid que permita el incremento del rendimiento de aplicaciones de alta productividad con un uso intensivo de datos de entrada. Este nuevo sistema tendrá las siguientes características:

- Toda su funcionalidad se proporcionará únicamente a través de las capacidades de metaplanificación siguiendo una filosofía "extremo a extremo". De este modo, su implantación no requerirá modificaciones en los distintos elementos de un sistema Grid.
- El sistema cache proporcionará la suficiente versatilidad para adaptarse a escenarios Grid heterogéneos.
- El sistema cache será diseñado para permitir una fácil implantación sobre cualquier tipo de metaplanificador.

Como segundo objetivo, el sistema cache se empleará en la mejora de la realización de dos aplicaciones eficientes de ajuste bioinformático. Estas aplicaciones se diseñarán desarrollando un novedoso método de aceleración de la exploración rotacional y serán convenientemente adaptadas a la ejecución en un sistema Grid:

1. ***ADP_EM***: Tipo de ajuste interno. Herramienta de ajuste de estructuras atómicas de proteínas dentro de mapas 3D de media-baja resolución.
2. ***FRODOCK***: Tipo de ajuste externo. Herramienta de predicción de la interacción tridimensionalmente de dos proteínas a partir de sus estructuras atómicas obtenidas de forma aislada.

Por último, empleando estas aplicaciones bioinformáticas se analizará la productividad y el comportamiento de un entorno Grid con el sistema cache propuesto. Además, se estudiará de forma más general la productividad proporcionada por el sistema cache empleando un conjunto de pruebas estándar de evaluación de Grids.

1.3 Estructura

La presente tesis doctoral se estructura en los siguientes seis capítulos:

- ***Introducción a computación Grid y aplicaciones bioinformáticas***: Introduce los conceptos principales de la computación Grid, describiendo sus características, las funcionalidades que proporciona y ejemplos de implementaciones. Debido a la importancia que en la presente tesis doctoral tiene el concepto de metaplanificador, se ha prestado especial importancia a su descripción. Posteriormente se indican aplicaciones de la tecnología Grid en el campo de la bioinformática. Por último, se detallan las características de las aplicaciones de ajuste a cuya mejora está enfocada la presente tesis doctoral.
- ***Sistema cache "Extremo a Extremo" sobre metaplanificación***: Realiza una descripción del diseño del sistema cache propuesto, indicando sus características así como las adaptaciones necesarias a los distintos escenarios presentes en un sistema Grid. El capítulo se cierra con la adaptación del sistema cache al metaplanificador GridWay. Esta adaptación será la empleada para realizar el análisis del sistema cache.
- ***Aplicaciones bioinformáticas de ajuste***: Describe un novedoso método de búsqueda de ajuste basado en el uso de representaciones mediante esféricos armónicos. Este método ha sido desarrollado y adaptado tanto a la resolución de problemas de ajuste multi-resolución (ADP_EM) como a la de problemas de ajuste proteína-proteína (FRODOCK). Asimismo, se describe la adaptación paralela de ambas aplicaciones para su uso sobre sistemas Grid.
- ***Validación de las aplicaciones de ajuste***: Este capítulo muestra los experimentos realizados para demostrar la validez y eficiencia de las dos aplicaciones diseñadas.
- ***Eficiencia del sistema cache***: Presenta y analiza los resultados que demuestran la mejora obtenida gracias al uso del sistema cache. Estas pruebas se han realizado empleando las aplicaciones paralelas diseñadas sobre una implantación del sistema cache en el metaplanificador GridWay. Además, también se muestran los resultados obtenidos con un conjunto de pruebas estándar para sistemas Grid que permiten determinar la mejora

proporcionada en problemas generales de alta productividad con amplia reutilización de datos de entrada.

- ***Conclusiones.***

2 Introducción a computación Grid y aplicaciones bioinformáticas

2.1 Computación Grid

Esta sección describe brevemente el paradigma de computación Grid empleado como marco de ejecución de las aplicaciones bioinformáticas de ajuste. En una primera sección se muestran las razones y necesidades que han llevado a la creación de este paradigma, así como una breve descripción de la evolución de la computación Grid. Una segunda sección describe las características que definen la computación en sistemas Grid. En la tercera sección, se describirá brevemente el paquete de herramientas Globus Toolkit, considerado como estándar *de facto* de la computación Grid, mostrándose su estructura y principales servicios. En las últimas secciones se hará especial hincapié en la funcionalidad de los servicios de metaplanificación en Grid, describiendo particularmente el servicio de metaplanificación GridWay.

Tipo de problema	Descripción
Supercomputación	Requieren una alta capacidad de cálculo. Aplicaciones bioinformáticas [1], de cálculo numérico o de estudio de fenómenos físicos y astrofísicos.
Sistemas de tiempo real	Tratamiento en tiempo real de flujos de datos a gran velocidad. Tiempo de respuesta crítico. Sistemas de control remoto de recursos dedicados, experimentos de física [2].
Procesado de datos intensivo	Tratamiento de conjuntos de datos de gran tamaño [3].
Entornos virtuales	Creación de entornos virtuales con capacidad de respuesta rápida. Teleinmersión [4].

Tabla 2.1- Tipos de problema con alto requerimiento computacional.

2.1.1 Motivación y evolución de la computación Grid

En las últimas dos décadas se ha producido un espectacular incremento de las capacidades ofrecidas por los computadores debido a los avances tanto en el campo del *hardware* como en la sofisticación del diseño de *software*. Sin embargo, continúan existiendo problemas que, debido tanto a los elevados requerimientos de cálculo como a la amplia y heterogénea cantidad de información manejada, no pueden ser resueltos en tiempos razonables empleando únicamente capacidades de supercomputadores individuales. Los distintos tipos de problemas de esta naturaleza se encuentran en diferentes campos y presentan distintas

Modelo de computación	Descripción
Computación sobre *Cluster*	Granja de equipos informáticos de parecidas características controlados de forma centralizada a través de un software de planificación (PBS [5]). Relación coste/rendimiento muy alta. Es una solución de ámbito local, de ampliación limitada, sólo adaptable sobre conjunto de equipos homogéneos y que conlleva costosas tareas de gestión y mantenimiento de los equipos.
Computación sobre Intranet	Agrupación de la capacidad computacional de los recursos hardware de una red de área local. Alto rendimiento con un coste muy bajo (LSF[6], CONDOR [7], SGE [8]). Solución de ámbito local, escalabilidad limitada.
Computación sobre Internet	Modelo cliente/servidor de acceso a recursos externos a través de Internet. Alto rendimiento de aplicaciones (SETI [9], BOINC [10]). Existencia de cuellos de botella en el acceso a servidores, bajo nivel de seguridad, ancho de banda de conexión limitado.

Tabla 2.2- Modelos de computación en red.

características (véase **Tabla 2.1**). La gran expansión de Internet junto a la accesibilidad a tecnologías de comunicación cada vez más rápidas y eficaces proporcionan la posibilidad de afrontar estos problemas mediante modelos de computación en red que permiten combinar recursos distribuidos y generalmente infrautilizados. Algunos de estos modelos son descritos en la **Tabla 2.2**. Si bien estos modelos permiten resolver problemas con grandes requerimientos computacionales de forma eficaz, adolecen de capacidad para integrar plenamente los diferentes recursos, no permitiendo mantener una administración general ni establecer políticas de planificación adecuadas. Así, surge la necesidad de un modelo que permita la coordinación y gestión adecuada de recursos computacionales heterogéneos y distribuidos no sólo geográficamente sino administrativamente. La tecnología que permite conectar sistemas distribuidos y heterogéneos, proporcionando las funcionalidades para emplear dichos sistemas como si de un único recurso computacional se tratara, se conoce como computación Grid [11]. El desarrollo de la computación Grid ha sido progresivo, identificándose habitualmente tres estadios o generaciones en su desarrollo [12]. Así, una primera generación surgida a principios de los años 90, se caracteriza por la aparición de proyectos enfocados a la conexión de diferentes elementos de supercomputación, dando lugar a una tecnología llamada metacomputación [13]. En estos primeros proyectos el énfasis recae en la conexión de diferentes recursos con el fin de proporcionar mayor capacidad computacional a aplicaciones de alta productividad, prestando atención a aspectos como la comunicación, gestión de los recursos y manipulación de datos remotos. Proyectos representativos de esta etapa fueron FAFNER [14] e I-WAY[15]. Posteriormente, las mejoras tecnológicas en las comunicaciones junto a la adopción de estándares permitieron, hacia finales de la década de los 90, la aparición de una nueva generación de sistemas Grid, cuya visión general vino dada por [11]. Esta nueva generación introduce el concepto de *middleware*. Un *middleware* es un conjunto de servicios que proporcionan a las aplicaciones acceso a los recursos computacionales. Estos servicios se estructuran en diferentes capas, proporcionando diferentes niveles de funcionalidad. Mediante el uso de un *middleware* se proporciona soporte a aspectos importantes de un Grid, tales como heterogeneidad, escalabilidad y adaptabilidad, ofreciendo la interoperabilidad necesaria para afrontar procesos computacionales a gran escala. Legion [16] o Globus [17] son tecnologías Grid creadas a partir de esta nueva definición, siendo Globus la tecnología *software*

que se ha convertido en el estándar *de facto* en la implantación de sistemas Grid. Por último, la última generación actualmente en desarrollo se caracteriza por un enfoque de los sistemas Grid orientado a un modelo de servicio y donde se presta especial importancia a la metainformación relativa a los sistemas. Mientras que en la generación anterior se hacía énfasis en las capacidades de computación y manejo de datos, la tercera y actual generación se centra en una mejor y más detallada descripción de los sistemas de forma que se facilite su automatización y dinamismo. Así, el enfoque con el que se estudian las infraestructuras Grid está más orientado a las aplicaciones que las emplean que a la visión de los recursos que la componen [18-19].

En la siguiente sección se detallan las características a las que esta evolución ha dado lugar, mostrándose las ventajas proporcionadas por este paradigma.

2.1.2 Características de la Computación Grid

La primera definición de la tecnología Grid fue dada por I. Foster y C. Kesselman [11]:

"A computable Grid is a hardware and software infrastructure that provides dependable, consistent, pervasive and inexpensive access to high-end computational capabilities"

Posteriores definiciones ampliaron este concepto. Así los mismos autores reformularon la definición [18]:

"Flexible, secure and coordinated resource sharing among dynamic collections of resources, individuals and institutions known as virtual organizations"

Y por último, la definición de tecnología Grid realizada para la segunda generación y comúnmente aceptada, dada por I. Foster en [20]:

"A Grid is a system that coordinates resources that are not subject to a centralized control using standard, open, general-purpose protocols and interfaces to deliver nontrivial qualities of services"

A partir de estas definiciones pueden establecerse las características generales que definen a un sistema Grid. Las tres principales características que deben ser contempladas a partir de la última definición son:

- ***Descentralización***: No debe existir un control centralizado sobre el Grid, de forma que no se limite el acceso a los recursos y se permita un alto grado de escalabilidad.

- ***Uso de protocolos generales y abiertos***: la coordinación de recursos heterogéneos sólo será posible empleando mecanismos generales aplicables en todo tipo de recursos y administraciones.
- ***Calidad de servicio no trivial***: Es decir, el uso combinado y coordinado de los diferentes recursos componentes de un Grid debe proporcionar capacidades superiores a las obtenidas por el uso individual de dichos recursos.

Otras características deseables en un sistema Grid y que en parte se derivan de las anteriores son:

- ***Heterogeneidad***: Un Grid estará compuesto de una amplia y variada tipología de recursos adscritos a diferentes tipos de administraciones.
- ***Escalabilidad***: Un sistema Grid deberá poder ampliarse de forma prácticamente ilimitada, siendo necesarios servicios para reducir latencias de comunicación.
- ***Seguridad***: Al ser empleados recursos adscritos a diferentes administraciones, se deberán proveer mecanismos para garantizar el correcto uso de dichos recursos. Del mismo modo, se deberá garantizar la confidencialidad de datos empleados por las aplicaciones.
- ***Adaptabilidad, flexibilidad y dinamismo***: Los diferentes recursos del sistema pueden cambiar sus prestaciones a lo largo del tiempo, presentar fallos en su funcionamiento e incluso ser inaccesibles temporalmente. Un sistema Grid deberá de estar provisto de mecanismos para asegurar la fiabilidad de las ejecuciones, adaptándose a las características de los recursos de forma dinámica. El hecho de emplear una política descentralizada y protocolos abiertos facilita la obtención de estas características.

Desde el punto de vista orientado a un modelo de servicios establecido en la tercera etapa del desarrollo de la tecnología Grid, un Grid puede ser descrito como diferentes conjuntos de servicios que conforman organizaciones virtuales (*Virtual Organizations*, VO) orientadas a proveer funcionalidad para resolver objetivos compartidos por diferentes administraciones. Estos servicios son entidades abstractas que proporcionan capacidades diversas, tales como capacidad de cálculo, almacenamiento, funcionalidades específicas, etc.

Para proporcionar todas las características descritas hasta ahora, un sistema Grid debe de estar estructurado en una serie de componentes que ofrezcan diferentes tipos de funcionalidades. Así, se ha definido una arquitectura general establecida en diferentes capas (véase **Fig. 2.1**), siguiendo el modelo de arquitectura empleado en Internet [18, 21]:

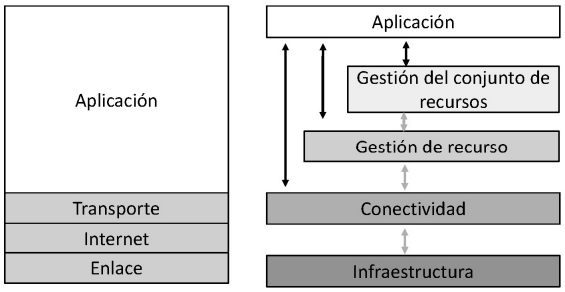

Fig. 2.1- Esquema de la arquitectura Grid comparada con la arquitectura Internet. A diferencia de esta última, en la arquitectura Grid las aplicaciones pueden interactuar con servicios de diferentes capas.

- ***Infraestructura***: Formada por todos los recursos computacionales que forman el sistema junto con la infraestructura de red que los une.
- ***Conectividad***: Formada por los servicios y protocolos que permiten la comunicación a bajo nivel entre los diferentes recursos tales como protocolos TCP/IP, SSL, etc. Estos deben de ser lo suficientemente amplios como para permitir la comunicación entre elementos heterogéneos y con distintas políticas de seguridad.
- **Gestión de recursos individuales**: Se engloban los servicios para el acceso a recursos individuales. Principalmente establecen los protocolos de información que permiten conocer, acceder y gestionar las características tanto estáticas (procesador, arquitectura, etc.) como dinámicas (carga de trabajo, memoria disponible, etc.) de un determinado recurso.
- ***Gestión de conjuntos de recursos***: Incluye los servicios que permiten emplear los recursos de forma conjunta. Entre los servicios necesarios se encuentran servicios de detección de recursos, planificación y asignación de tareas, monitorización y gestión de bases de datos.
- ***Aplicaciones***: Engloba las aplicaciones que pueden ser realizadas en el entorno Grid.

A diferencia de la arquitectura Internet donde la funcionalidad de cada capa queda completamente envuelta por la capa superior, en esta arquitectura las aplicaciones tienen acceso directo al resto de capas. De este modo, por ejemplo, una aplicación puede emplear servicios de la capa de gestión individual de recursos para transferir información a un determinado recurso computacional.

Esto implica una alta complejidad en el adecuado acceso al sistema Grid por parte de las aplicaciones. Sin embargo, el uso de metaplanificadores permite obviar esta compleja accesibilidad a servicios en distintas capas. De forma simplificada, un metaplanificador es un servicio que gestiona todas las operaciones necesarias para la realización adecuada de las tareas solicitadas por usuarios. Para ello, interactúa con todos los niveles de capas. De este modo, el uso de un metaplanificador introduce una nueva capa que actúa como interfaz entre las aplicaciones de usuario y el resto del sistema Grid, facilitando el uso de sus servicios (**Fig. 2.2**). Los metaplanificadores son, por tanto, elementos de gran importancia en el acceso a un sistema Grid. En este sentido y como se mostrará posteriormente, un metaplanificador proporciona una funcionalidad esencial que facilitará la implantación eficaz del sistema cache propuesto.

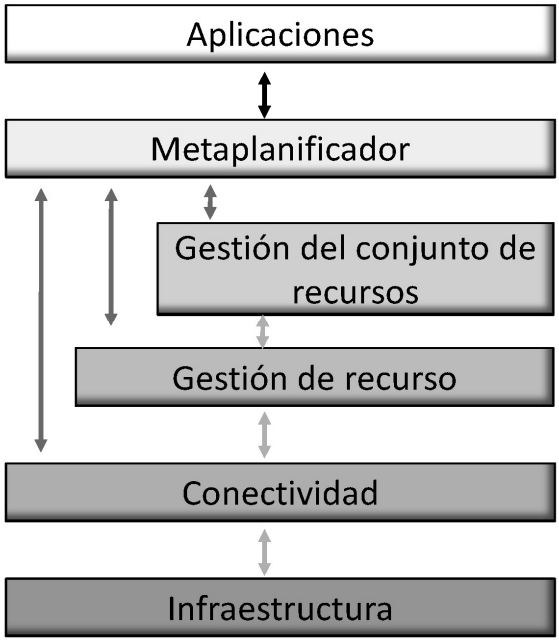

Fig.2.2- Uso de un metaplanificador en arquitectura Grid. El metaplanificador actúa como interfaz, ocultando la complejidad en el uso de los diferentes servicios Grid a los usuarios.

2.1.3 Globus Toolkit

Si bien existen diferentes implementaciones de la tecnología Grid [22-24], es la implementación Globus Toolkit (GT), desarrollada por el proyecto Globus [25], la más empleada dentro de la comunidad Grid y considerada como estándar *de facto*. GT es un *software* de código abierto y organizado como un conjunto de elementos débilmente acoplados que proporcionan las diferentes funcionalidades de un sistema Grid. La versión 4 de GT sigue diferentes especificaciones que permiten la implementación de aplicaciones sobre recursos de tipo heterogéneo (**Fig. 2.3**):

Fig.2.3- Arquitectura de especificaciones seguida por Globus.

	Componente	Funciones
Seguridad	**GIS**: *Grid Infrastructure Security* [26]	Autentificación comunitaria a VOs Autentificación y autorización Delegación Gestión de credenciales
Gestión de Datos	**GridFTP**: *Grid File Transfer Protocol* [27-29]	Transferencia segura y confiable de datos entre recursos
	RFT: *Reliable File Transfer* [8, 30]	Servicio de transferencias seguro
	RLS: *Data Location Service*	Localización de réplica de datos en VOs
	DRS: *Data Replication Service*	Gestión de réplicas de datos
	OGSA-DAI: *OGSA-Data Access Integration*	Acceso e integración de datos en distintos formatos
Gestión de Ejecución	**GRAM**: *Grid Resource Allocation and Management* [31]	Control y monitorización de ejecuciones remotas
	CSF: *Community Scheduler Framework* [32]	Interfaz común para planificadores de recursos
	Management Workspace	Gestión dinámica de espacios de trabajo en recursos
	GridWay	Metaplanificación eficiente sobre recursos
Servicios de Información	**MDS:** *Monitoring and Discovery Service* [33]	Indexación de recursos Gestión de eventos Interfaz web de información a usuarios
Componentes comunes	**Soporte Java, C, Python**	Funcionalidad para servicios sobre distintos lenguajes

Tabla 2.3- Componentes de Globus Toolkit versión 4.

- ***Servicios Web***: Permiten implementar aplicaciones cliente/servidor débilmente acopladas de forma independiente a la plataforma de ejecución. Para ello se establecen mecanismos de comunicación basados en el uso del protocolo HTTP (*HyperText Transfer Protocol*).
- ***OGSA***: Acrónimo de *Open Grid Services Architectures*, es una especificación de arquitectura de servicios en Grid establecida por el *Global Grid Forum* [34]. Esta especificación define una arquitectura abierta, común y estándar para todas las aplicaciones basadas en sistemas Grid. Esto se consigue mediante el establecimiento de interfaces y requisitos estandarizados para una serie de servicios establecidos. Estos servicios son:
    - *Servicio de gestión de VO*: Proporciona gestión de usuarios y recursos de las diferentes VO.

- *Servicio de gestión y descubrimiento de recursos*: Proporciona a las aplicaciones accesibilidad a los recursos adecuados.
- *Servicio de gestión de trabajos*: Gestiona la ejecución de aplicaciones.

- ***WSRF***: Acrónimo de *Web Services Resource Framework*, establece una infraestructura para crear los servicios establecidos por la especificación OGSA mediante servicios Web.

Siguiendo estas especificaciones, GT4 está compuesto por diferentes servicios agrupados en distintas áreas tal y como se muestra en la **Tabla 2.3**.

2.1.4 Metaplanificación

Desde el punto de vista de un usuario, uno de los aspectos más importantes en la gestión de un entorno Grid es la planificación y ejecución de aplicaciones o tareas dentro de éste. Si bien implementaciones Grid como Globus proporcionan servicios que pueden ser empleados directamente para lanzar aplicaciones, su adecuado uso requiere cierto conocimiento de las características del Grid junto a un esfuerzo adicional por parte del usuario. Dado que, en muchos casos, los potenciales usuarios de un Grid no poseerán suficiente conocimiento para emplear estos servicios de forma adecuada, la implementación de herramientas que faciliten su acceso resulta de gran importancia. Por tanto, un adecuado diseño de estas herramientas, llamadas metaplanificadores, es crítico. El usuario, al emplear un metaplanificador, debe de percibir que los recursos del entorno son aprovechados de forma óptima, sin tener que preocuparse de cómo éstos son seleccionados y empleados.

La planificación de tareas en un entorno Grid es una labor compleja. Esta planificación se define como el proceso de toma de decisiones acerca de qué recursos de un conjunto heterogéneo y perteneciente a diferentes administraciones deben ser empleados para realizar diferentes tareas [35]. Esto lleva asociado una serie de dificultades derivadas de la naturaleza del propio sistema Grid:

- ***Falta de autoridad y control sobre recursos***: Debido a que los recursos están bajo el control de administraciones locales se impone la necesidad de establecer una negociación para emplear temporalmente y de forma limitada los recursos.

- ***Comportamiento dinámico***: los recursos pueden cambiar sus prestaciones e incluso inhabilitarse temporalmente, por lo que la planificación debe poder adaptarse a estos cambios dinámicos.
- ***Entorno competitivo***: Los recursos de un sistema Grid son empleados por múltiples usuarios. Un planificador debe estudiar la carga de trabajo de los recursos a la hora de emplearlos.

Por estas razones, un planificador Grid debe de ser capaz de establecer las necesidades de una tarea y, teniendo en cuenta la accesibilidad, prestaciones y carga de trabajo puntuales de los recursos disponibles, determinar qué recursos debe de reservar y emplear para la realización de la mencionada tarea.

No existe una única forma de realizar una planificación de este tipo y la máxima prestación de los recursos no puede ser garantizada. Sin embargo, un aspecto fundamental en la planificación de tareas, que siempre tendrá que ser estudiado, concierne a la obtención de información acerca del Grid. Mientras en otros entornos es posible disponer de un conocimiento pleno, en el caso de un sistema Grid esto no es posible debido a su naturaleza dinámica. Así, la calidad de la planificación desarrollada dependerá de la cantidad y calidad de la información disponible. Para obtener esta información los metaplanificadores han de interaccionar con sistemas de gestión de información proporcionados por el sistema Grid (En GT, por ejemplo, el servicio MDS). Por tanto, un metaplanificador tendrá que realizar una serie de pasos básicos. En primer lugar, se ha de determinar aquel conjunto de recursos disponible en el entorno que resulte adecuado a las necesidades de cada tarea. En segundo lugar una decisión

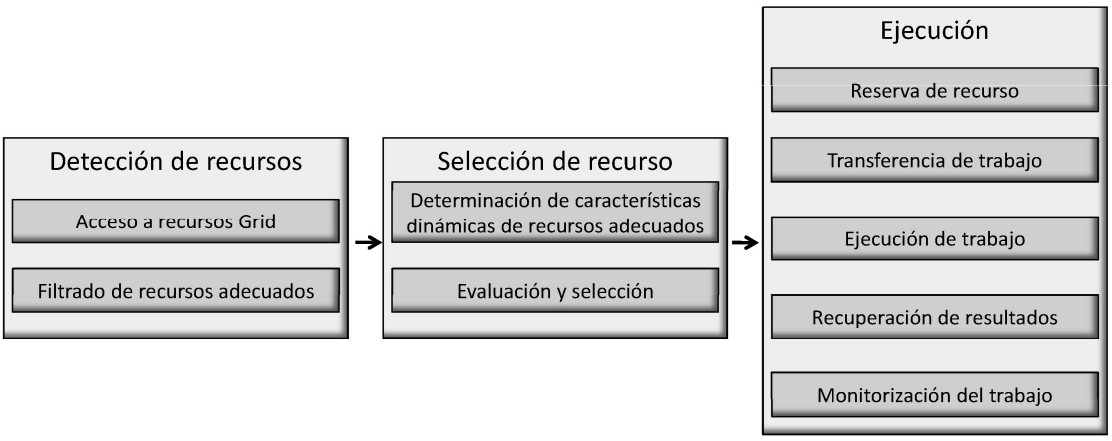

Fig.2.4- Fases en la planificación Grid para la ejecución de un trabajo.

sobre qué recursos de este conjunto deben utilizarse es tomada. Por último, se realiza la ejecución de la tarea sobre los recursos manteniendo una monitorización para asegurar un resultado correcto. Cada uno de estos pasos requiere la realización de acciones concretas (**Fig. 2.4**):

- ***Detección de recursos***: Primeramente es necesario determinar los recursos accesibles y cuyo uso por parte del usuario esté autorizado. Posteriormente, una especificación mínima de los requerimientos necesarios para la tarea por parte del usuario debe de ser estudiada. Los requerimientos a establecer pueden cubrir una amplia gama de características tales como sistema operativo, memoria RAM disponible, bases de datos accesibles, etc. Por último, se realiza un filtrado del subconjunto de recursos accesibles a través de los requerimientos indicados, obteniendo el subconjunto de recursos que permite realizar la tarea.
- ***Selección***: Para realizar la asignación de recursos de forma óptima, se establece una consulta acerca de la información dinámica del subconjunto de recursos. Esta información dinámica incluye aspectos como carga actual de trabajo, tiempo de respuesta, etc. Resulta importante que esta consulta cubra el mayor número de aspectos posible. Cuanto mayor y heterogénea sea esta información, más fácil será tomar una decisión adecuada acerca de qué recursos seleccionar para la ejecución de la tarea.
- ***Ejecución de tarea***: La falta de control sobre los recursos complica la ejecución de una tarea sobre estos. Inicialmente se ha de establecer una reserva de los recursos mediante una negociación con la administración local. Una vez hecho esto se realiza el envío a los recursos de los datos necesarios y se modifica el entorno del recurso para adecuarlo a la ejecución. Por último, se realiza la ejecución y, una vez finalizada, se recuperan los datos de salida, eliminando dichos datos de los recursos y restableciendo la configuración inicial de estos. Al mismo tiempo que todos estos pasos, operaciones de chequeo y monitorización son realizadas para comprobar la correcta ejecución. En caso de que un fallo evite su realización, la tarea será replanificada en un recurso diferente.

Como se puede comprobar, el conjunto de acciones necesarias para realizar una planificación eficiente de una tarea en un entorno Grid hace de ésta una operación de gran complejidad. Especialmente difícil resulta la elección de los recursos más adecuados de entre los disponibles. Debido a la naturaleza del entorno Grid se hace imposible determinar qué elección acarreará la ejecución más óptima posible. Únicamente gestionando eficientemente una amplia variedad de información dinámica acerca de los diferentes recursos se puede garantizar una aproximación a esta ejecución óptima.

2.1.4.1 Ejemplos de Metaplanificadores

En esta sección se describen brevemente algunos de metaplanificadores creados para interactuar con los *middlewares* más habitualmente empleados.

CSF

El metaplanificador CSF (del inglés *Community Scheduler Framework*) [32] ha sido desarrollado por *Platform Computing* junto con la Universidad Jili de China. CSF es un metaplanificador de código abierto que realiza tareas de planificación mediante el uso de servicios proporcionados por Globus. CSF permite realizar reserva de recursos y establecer planificaciones simples de ejecuciones sobre gestores locales de recursos tales como LSF, PBS o SGE. En **Fig. 2.5** se muestra la arquitectura del metaplanificador, donde se indican los tres servicios básicos:

- ***Servicio de trabajos***: Gestiona las ejecuciones demandadas por el usuario.
- ***Servicio de Reservas***: Gestiona la disponibilidad de recursos del Grid.
- ***Servicio de Planificación***: Define políticas de planificación a nivel de VO y a diferentes niveles de gestión de recursos.

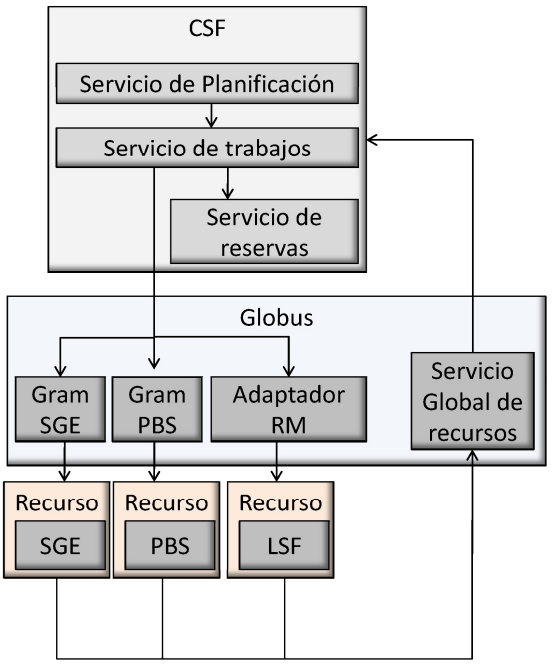

Fig.2.5- Arquitectura CSF.

GSB

El metaplanificador GSB (del inglés *Grid Service Broker*) [36] permite la planificación de tareas sobre

diferentes *middlewares* como Globus, Alchemi [37] o Unicore [23] . GSB permite establecer límites de tiempo en la ejecución de tareas y gestiona errores de ejecución [38]. En la **Fig. 2.6** se muestra la arquitectura del metaplanificador. La principal diferencia con respecto a CSF radica en el gestor de *middleware* que permite al usuario ejecutar tareas de forma transparente al *middleware* empleado.

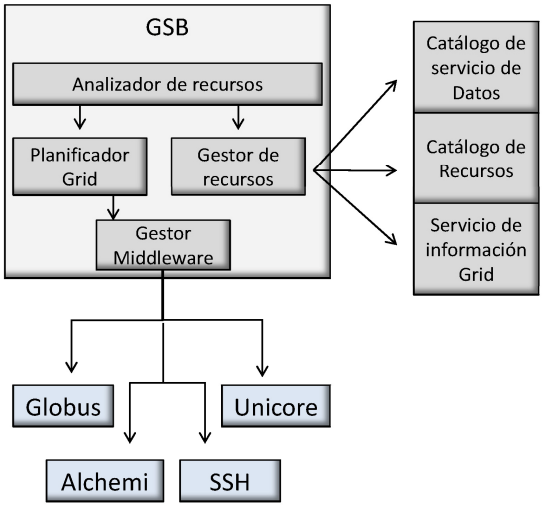

Fig.2.6- Arquitectura GSB.

GRMS

GRMS (del inglés *GridLab Resource Management System*) [39] es un metaplanificador incluido en el *middleware* GridLab [40]. Las principales características que presenta son la inclusión de un lenguaje de descripción de tareas propias, capacidad para ejecutar tareas con dependencias y migración de tareas entre recursos. Su estructura (véase **Fig. 2.7**) está compuesta por los siguientes modulos:

- ***Gestor de Flujo de Trabajos***: Encargado de la recepción de tareas del usuario.
- ***Broker***: Encargado de dirigir la ejecución de tareas determinando en qué recursos deben realizarse.

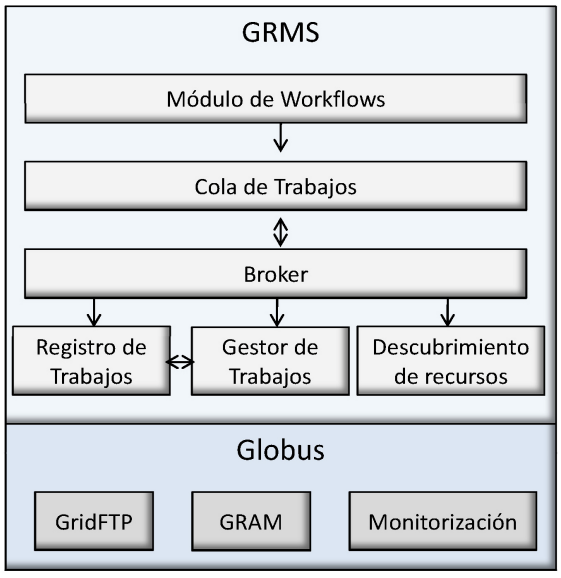

Fig.2.7- Arquitectura GRMS.

WMS

WMS (del inglés *Workload Manager System*) [41] es el sistema de envío y gestión de tareas del *middleware* GLite [42], empleado en la infraestructura EGEE (*Enabling Grids for E-sciencE*) [43]. WMS permite realizar el tratamiento de las tareas siguiendo dos políticas distintas:

- ***Política ansiosa***: Una tarea es asignada a un recurso inmediatamente después de ser recibida la petición.
- ***Política perezosa***: La asignación de las tareas es retrasada hasta que un recurso se encuentra disponible. Cuando existen varios recursos disponibles se realiza una evaluación para determinar el más adecuado para la tarea.

Para implementar la segunda política, el módulo central WM (del inglés *Work Manager*) emplea tres módulos:

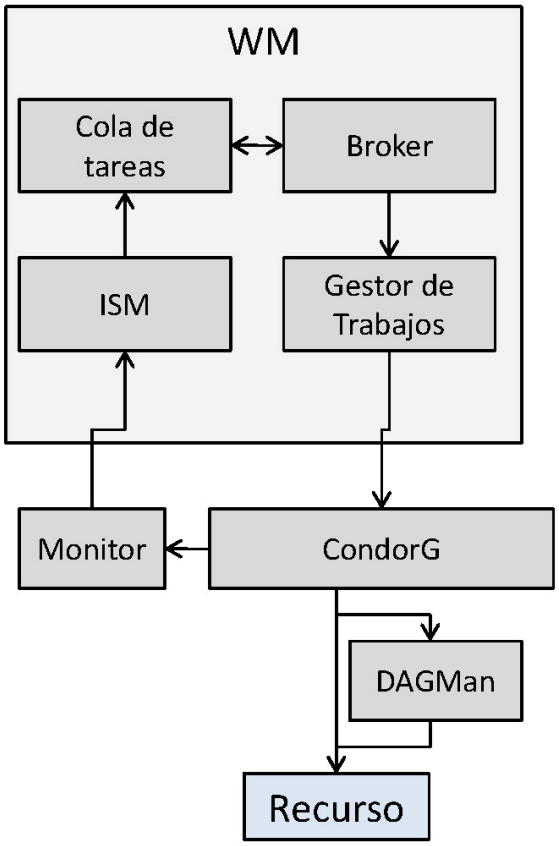

Fig.2.8- Arquitectura WMS.

- ***ISM (Information Supermarket)***: Es un repositorio de información acerca de los recursos del sistema Grid.
- ***Cola de tareas***: Almacena las peticiones realizadas al sistema. Estas peticiones son especificadas mediante un lenguaje descriptor JDL (*Job Description Language*) que permite indicar las características y requerimientos de la tarea.
- ***Broker***: Empleando la información almacenada tanto en la cola de tareas como en el repositorio ISM establece la asignación de tareas a recursos.

Por último, el sistema realiza el lanzamiento de las tareas mediante el servicio CondorG [44] y permite la realización de conjuntos de tareas con dependencias apoyándose en el uso del metaplanificador DAGMan [45]. En la **Fig. 2.8** se muestra un esquema de su estructura.

2.1.4.2 GridWay

GridWay [46-47] es un metaplanificador desarrollado sobre Globus (aunque también puede ser empleado sobre GLite) que afronta la ejecución adecuada de

tareas sobre entornos Grid considerando su naturaleza dinámica mediante la implementación de:

- ***Planificación adaptativa***: Los recursos accesibles son periódicamente chequeados para determinar diferentes características dinámicas (carga de trabajo, rendimiento medio, etc.). La planificación se verá alterada en función de esta información.
- ***Ejecución adaptativa***: Con el fin de obtener un rendimiento óptimo así como para permitir tolerancia a fallos se permite la migración de tareas de unos recursos a otros.

GridWay permite una ejecución simple y eficiente de tareas sobre entornos dinámicos Grid mediante una filosofía "enviar y olvidar". El núcleo del metaplanificador GridWay está formado por un agente de envíos encargado de realizar todas las acciones de planificación descritas anteriormente. Además, realiza operaciones de control para determinar la correcta ejecución de las tareas enviadas. La adaptación a las condiciones dinámicas del entorno se obtiene mediante la planificación adaptativa. Una vez que una determinada tarea es asignada a un recurso, ésta migrará a un recurso distinto si el rendimiento obtenido es muy bajo o si se detectan errores en el recurso. El rendimiento de una tarea es evaluado periódicamente aplicando un programa de medición de la degradación del rendimiento y determinando el tiempo total de suspensión de la tarea en el recurso. Mediante rutinas de búsqueda de recursos se establece una lista de recursos adecuados para la tarea, de forma que puede determinarse si es conveniente la migración a un recurso diferente.

En la siguiente sección se detalla la arquitectura del metaplanificador GridWay. Posteriormente, se describen las acciones realizadas por el metaplanificador durante la ejecución de una tarea. Por último, se resumen las ventajas que se aprecian en el uso de GridWay frente a otros planificadores y que han llevado a emplearlo en el desarrollo de la presente tesis doctoral.

2.1.4.2.1 Arquitectura

Fig. 2.9 muestra un esquema de los diferentes módulos que componen el metaplanificador GridWay. Como ya se ha indicado, el elemento principal es un agente de envíos, ejecutado locamente desde el nodo de cómputo del usuario, formado por los siguientes componentes:

- ***Gestor de peticiones***: Permite al usuario realizar peticiones de ejecución de tareas a través de una API (*Application Programming Interface*). El cliente también puede realizar operaciones de control, tales como eliminar o bloquear una determinada tarea.

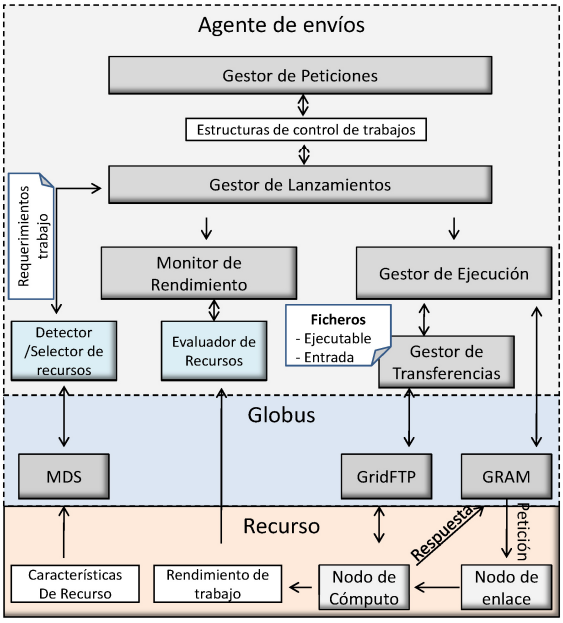

Fig.2.9- Arquitectura GridWay.

- ***Gestor de lanzamientos***: Es ejecutado periódicamente y determina la ejecución de tareas pendientes o que deben ser replanificadas. El gestor de lanzamientos invoca la ejecución de un detector y de un selector de recursos para cada tarea, los cuales devuelven una lista de posibles recursos a emplear. Empleando esta lista, el gestor de lanzamientos invoca al gestor de ejecución para cada tarea.
- ***Gestor de ejecución***: Es el encargado de la ejecución de tareas en un determinado recurso. Para ello, interactúa con diferentes servicios Globus y con el gestor de transferencias. El gestor de ejecución controla todo el ciclo de vida de cada tarea en un recurso hasta que ésta finalice o sea cancelada.
- ***Gestor de transferencia***: Realiza, mediante la interacción con servicios Grid, el envío y recepción de ficheros desde la máquina local a los recursos del entorno donde son ejecutadas las tareas.
- ***Monitor de rendimiento***: Periódicamente es activado para determinar el rendimiento de las tareas en ejecución. Para cada tarea es activado un evaluador de rendimiento. Si éste devuelve un rendimiento muy bajo la tarea es seleccionada para ser replanificada.

2.1.4.2.2 Acciones de planificación

Desde el momento en que un usuario pide la realización de una tarea hasta que ésta finaliza el metaplanificador realiza una serie de acciones que definen su ciclo de vida.

Detección y selección de recursos

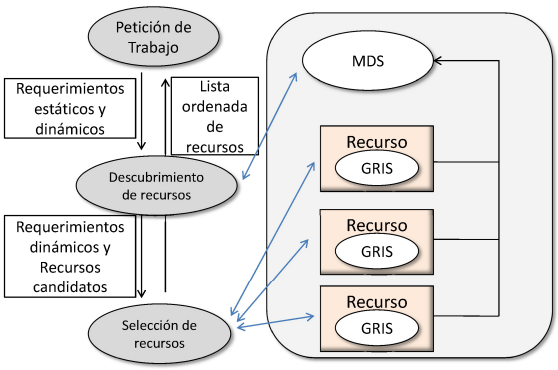

Fig.2.10- Esquema de detección y selección de recursos en GridWay.

Cuando una tarea es requerida por parte del usuario, entra en una cola de tareas pendientes. El gestor de lanzamientos realiza el llamamiento a un detector de recursos indicando los requerimientos que el usuario haya establecido para la tarea. Los requerimientos que se emplean para la selección de recursos son de naturaleza estática (sistema operativo, arquitectura, etc.). Empleando estos requerimientos estáticos el detector de recursos filtra aquellos disponibles mediante la comunicación con el servidor MDS de GT.

Posteriormente, tras una comprobación de que los recursos son realmente accesibles, el selector de recursos es ejecutado para determinar la prioridad entre los recursos filtrados en función de requerimientos dinámicos (Espacio en disco, carga de trabajo, etc.) fijados por el usuario. Para determinar si se cumplen estos requerimientos dinámicos, el selector de recursos interacciona con los servidores GRIS (del inglés *Grid Resource Information Service*) de cada recurso. Básicamente, el selector de recursos priorizará el uso de aquellos que, cumpliendo los requerimientos dinámicos, minimicen el tiempo de respuesta para la tarea. Esta política "avariciosa" es efectiva si bien no garantiza productividad máxima [48]. La lista priorizada de recursos obtenida es empleada por el gestor de lanzamientos para la ejecución de la tarea (ver **Fig. 2.10**).

Esta aproximación modular garantiza la extensibilidad de la selección de recursos, pudiéndose emplear diferentes estrategias.

Preparación del sistema remoto (fase prolog)

Una vez el gestor de lanzamientos determina la ejecución de una tarea en un recurso, el gestor de ejecución realiza las operaciones necesarias para llevar a cabo dicha ejecución. La primera acción realizada por el gestor de ejecución es la adecuación del recurso remoto a la ejecución (fase *prolog*). En la fase *prolog* se establece el directorio de trabajo en el recurso remoto y tanto el ejecutable de la tarea como los ficheros de entrada son transferidos a dicho directorio. El establecimiento del directorio de trabajo remoto se realiza a través del servicio GRAM de GT, permitiendo además la transferencia de una rutina de control de la

tarea remota. Los ficheros por su parte, son transferidos mediante el gestor de transferencias empleando servicios Grid como GridFTP. Sin embargo, esta transferencia sólo es posible si el nodo de cómputo remoto empleado es accesible de forma directa. En sistemas de recursos cerrados, el acceso se realiza a través de un nodo de acceso (*front-end*), imposibilitándose el acceso directo. Por este motivo, GridWay establece diferentes esquemas de transferencia. En caso de poderse establecer un acceso directo al espacio de almacenamiento del nodo de cómputo, el gestor de lanzamientos realiza la transferencia siguiendo un esquema directo, iniciándose la comunicación del origen al destino. En caso contrario, se establece un esquema de transferencia inverso en el que las operaciones de trasferencia se delegan a la siguiente fase (*wrapper*) y la comunicación se inicia desde el destino (el nodo de cómputo una vez elegido) al origen.

Ejecución de la tarea (fase wrapper)

Tras la fase *prolog* el gestor de ejecución realiza el lanzamiento de la tarea en el recurso remoto. Este lanzamiento se gestiona a través de la rutina de control remoto transferida en la fase anterior. Tal y como se ha indicado, si la transferencia de ficheros de entrada no pudo ser realizada en la fase anterior por no encontrarse los nodos de cómputo accesibles directamente, el esquema de transferencia inverso es aplicado. En tal caso, en primer lugar se realiza la petición, a través de los servicios de Globus, de la transferencia de los ficheros necesarios que se encuentren en el recurso local del usuario. Posteriormente el ejecutable de la tarea es invocado comenzando así la ejecución. Cuando la tarea finalice, si se aplica el esquema de transferencia inverso, la rutina realizará la transferencia de los ficheros de salida resultantes.

Recuperación de datos de salida (fase epilog)

Cuando el esquema de transferencia es directo y se ha producido la finalización de la tarea, el gestor de ejecución realiza la transferencia de los ficheros de salida invocando al gestor de transferencias. Por último, el nodo de cómputo es restablecido a su estado inicial, eliminando la información de la tarea realizada.

Migración de tareas

Durante las fases *prolog* y *wrapper* una tarea puede ser replanificada, de forma que el gestor de ejecución fuerza la fase *epilog* eliminando la tarea del recurso remoto al que estaba adscrito y pasando su gestión de nuevo al gestor de lanzamientos. Las razones por las que una tarea es replanificada pueden ser de distinto tipo:

- ***Migración forzada por usuario***
    - El gestor de peticiones recibe una petición de replanificación de la tarea por parte del usuario.
- ***Migración forzada por el metaplanificador***
    - El monitor de rendimiento detecta una baja productividad o un tiempo de espera en la ejecución de la tarea muy alto.
- ***Migración forzada por el recurso Grid***
    - El gestor de ejecución detecta la cancelación de la ejecución de la tarea por parte de la administración local del recurso.
    - Se producen errores de comunicación entre el gestor de ejecución y el recurso, imposibilitándose la correcta ejecución de la tarea.

Monitorización de rendimiento

Durante la realización de todas las acciones anteriores el monitor de rendimiento invoca periódicamente a evaluadores de rendimiento para cada tarea en ejecución. El tipo de evaluador empleado es elegido por el usuario, pudiendo emplear en la evaluación del rendimiento mecanismos de lógica difusa, comparación de rendimiento actual con rendimiento inicial o límite mínimo de rendimiento aceptable.

2.1.4.2.3 Ventajas de GridWay

GridWay proporciona una forma simple de lanzamiento de trabajos. Además, aglutina las principales ventajas mostradas por los metaplanificadores indicados anteriormente. Las características que hacen de GriWay un metaplanificador eficaz son:

- ***Arquitectura fiable y extensible***: GridWay al estar diseñado de manera modular, permite la adaptación a diferentes infraestructuras como EGEE, TeraGrid[49], OSG [50] o NorduGrid[51].
- ***Soporte estándar***: El metaplanicador GridWay implementa estándares OGF (del inglés *Open Grid Forum*) [52] como el interfaz de programación DRMAA (*Distributed Resource Management Application API*) [53], para distintos lenguajes de programación como C, Java o Perl. De esta manera, asegura la compatibilidad para aplicaciones que se ejecuten en recursos que implementen dicho estándar. Además, permite la ejecución,

monitorización, sincronización y control de trabajos usando el estandar JSDL [54] de OGF.

- ***Control del flujo***: Permite a los usuarios ejecutar conjuntos de tareas con dependencias entre ellas.
- ***Ejecución dinámica y adaptativa***: El metaplanificador GridWay realiza migración de tareas permitiendo de esta manera que la ejecución de los trabajos se adapte al estado del Grid.
- ***Fácil implantación***: GridWay no requiere servicios adicionales por parte de los sistemas Grid, apoyando su funcionalidad en servicios ya establecidos. De este modo, únicamente se requiere el establecimiento del software GridWay sobre el recurso del usuario.

2.2 Bioinformática y computación Grid

La bioinformática es una disciplina que estudia la aplicación de las ciencias de la computación en el campo de la biología. En ella se abarcan aspectos tales como la creación y gestión de bases de datos, diseño de algoritmos y técnicas estadísticas o de manejo y análisis de datos biológicos. De este modo, la bioinformática combina adecuadamente el conocimiento de materias como matemáticas, estadística, biotecnología, medicina o ciencias de la computación para obtener una mejor compresión de los sistemas vivos. De forma general, las técnicas resultantes de esta combinación requieren un uso intensivo tanto de datos como de capacidades computacionales. En este sentido, la tecnología Grid proporciona un marco adecuado para afrontar la realización de diferentes aplicaciones bioinformáticas.

La combinación de aplicaciones bioinformáticas con la tecnología Grid ha sido ampliamente desarrollada durante los últimos años [55]. De este modo existen aplicaciones adaptadas a la tecnología Grid en áreas tan diversas como el alineamiento de secuencias [56], genómica [57], investigación oncológica [58], ingeniería molecular [59] o reconstrucción tridimensional de estructuras moleculares [60]. Además, existen diversos proyectos con el fin de proporcionar entornos Grid adecuados para permitir el desarrollo de estudios bioinformáticos. Algunos de estos proyectos son BioinfoGRID [61], OpenMolGRID [62], myGrid [63] y VLe [64]. También son relevantes los proyectos Genome@home [65], Folding@home [66] y FightAIDS@home [67] que permiten emplear tiempo de

computo en recursos computacionales cedidos de forma gratuita por usuarios de todo el mundo para, respectivamente, el diseño de nuevas proteínas, el estudio del plegamiento de proteínas o la investigación sobre nuevas drogas contra el SIDA.

De entre las diversas áreas bioinformáticas, el modelado de moléculas y complejos macromoleculares a partir del ajuste de información estructural presenta características que lo hacen idóneo para la utilización de la tecnología Grid. La realización de complejos cálculos para evaluar cada ajuste, junto a la necesidad de realizar dichos ajustes sobre extensas colecciones de estructuras, hacen a este tipo de aplicaciones computacionalmente costosas. Además, presentan características propias de la computación de alta productividad. Básicamente, una aplicación de alta productividad se caracteriza por la ejecución de un alto número de instancias de un mismo cálculo pero con distintos parámetros de entrada. Esta característica hace que este tipo de aplicaciones pueda ser fácilmente implementado sobre tecnología Grid, ya que es posible realizar una paralelización de los cálculos sobre los extensos recursos computacionales proporcionados por dicha tecnología. Un ejemplo del uso de tecnología Grid en la resolución de problemas de ajuste puede verse en [68] donde se integra el programa de ajuste DOCK [69] sobre un entorno Grid para realizar el estudio de diseño de drogas. También ha sido empleado una aplicación de ajuste sobre Grid en [70], orientado a la búsqueda de nuevas drogas contra la malaria. La siguiente sección muestra en mayor detalle la problemática asociada a este tipo de aplicaciones bioinformáticas, describiendo brevemente la metodología que se empleará en la presente tesis doctoral para su resolución.

2.2.1 Métodos de ajuste tridimensional y aplicaciones bioinformáticas

La búsqueda del mejor ajuste entre distintos objetos es un problema ampliamente estudiado principalmente en el área del reconocimiento visual. En el reconocimiento de formas bidimensionales pueden encontrarse aproximaciones que emplean el ajuste de contornos [71-72] o una estructuración jerarquizada de las formas [73]. De forma similar, en el reconocimiento y clasificación de formas tridimensionales pueden emplearse distintas caracterizaciones [74]. Los métodos de reconocimiento de formas tridimensionales pueden ser clasificados según empleen una descripción basada en las características de las formas (distancias,

ángulos, distribución espacial) [75-77], una representación jerarquizada en grafos [78] o una representación geométrica obtenida a partir de múltiples vistas a distintos ángulos [79].

Si bien existen otras alternativas [80-81], la metodología más empleada para el ajuste de información estructural en el ámbito de la bioinformática es similar a la del reconocimiento de formas mediante caracterización de la distribución espacial [82]. Siguiendo esta línea, el ajuste de información estructural se basa en la búsqueda de la máxima correlación entre caracterizaciones de estructuras mediante una exploración de distintos desplazamientos y rotaciones entre los objetos.

En las siguientes secciones se introducen los principales conceptos que definen esta metodología. En una primera sección se indican los distintos tipos de ajustes que pueden ser estudiados, posteriormente se indican los distintos criterios con los que un ajuste es evaluado. En la tercera sección se describen las aproximaciones mediante las que puede afrontarse la exploración de posibles ajustes. Por último, se describen aquellas aplicaciones bioinformáticas en las que esta metodología de ajuste puede ser empleada.

2.2.1.1 Tipos de ajuste

El objetivo de un método de ajuste tridimensional consiste en encontrar la orientación u orientaciones relativas entre dos o más objetos tridimensionales que maximicen una determinada relación entre ellos. La naturaleza de esta relación determina el tipo de ajuste a realizar. **Fig. 2.11** ilustra esquemáticamente los

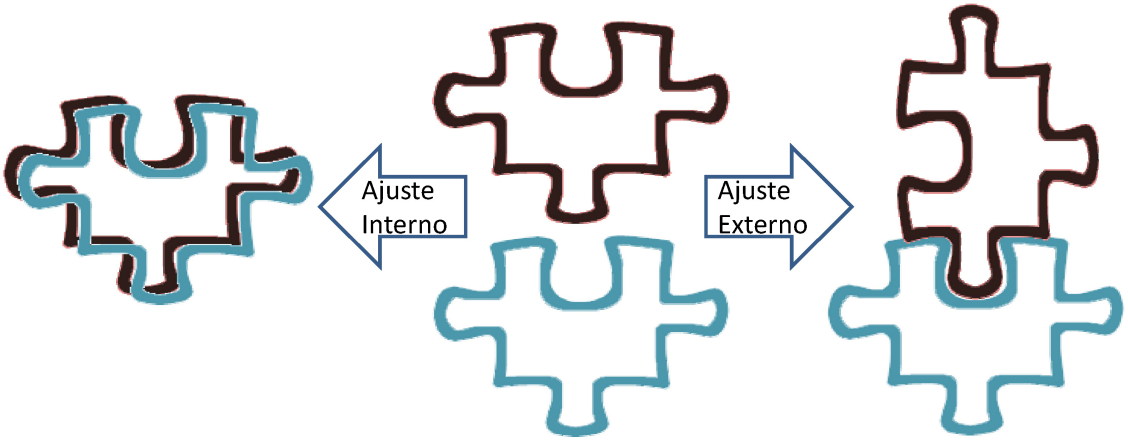

Fig.2.11- Tipos de ajuste. El tipo de ajuste interno busca maximizar la superposición entre objetos. El tipo de ajuste externo, por el contrario, determina la mejor superficie de contacto entre ellos.

tipos de ajuste que pueden realizarse. Así, el ajuste es de tipo interno cuando se busca la mejor superposición de una propiedad de los objetos. Por el contrario, un método de ajuste externo se orienta a la búsqueda del mejor contacto o complementariedad entre las superficies de los objetos, encajándose éstos como si de piezas de un puzle se tratara.

2.2.1.2 Criterios de ajuste

De forma general, un método de ajuste realiza una exploración del espacio de posibles posicionamientos entre objetos. La bondad del ajuste para cada uno de estos posicionamientos relativos se evalúa mediante una función o criterio de ajuste que generalmente consiste en el cálculo de correlación para una o varias propiedades de los objetos. Esta función de correlación dependerá del tipo de ajuste que se realice así como del tipo de propiedad empleada.

De este modo, en el caso de un ajuste interno cuyo objetivo es maximizar la superposición de una propiedad λ de dos objetos tridimensionales A y B, dado un determinado posicionamiento P para el objeto B con respecto a A y siendo M_λ^A y M_λ^B las matrices o mapas tridimensionales donde está proyectada dicha propiedad para cada objeto, la evaluación del posicionamiento se puede obtener simplemente mediante un producto escalar:

$$C(P) = \int M_\lambda^A \cdot \Lambda_P M_\lambda^B = \sum_x^X \sum_y^Y \sum_z^Z M_\lambda^A[x,y,z] \cdot M_\lambda^B[\Lambda_P(x,y,z)] \qquad (2.1)$$

Donde Λ_P es un operador que modifica rotacional y traslacionalmente las coordenadas de un objeto tridimensional con respecto al otro que permanece fijo. *X*, *Y*, y *Z* son las dimensiones cartesianas.

Para un ajuste externo el criterio de ajuste se orienta a la maximización de la complementariedad entre las superficies de contacto. Para ello, valores positivos de un mapa son correlacionados con regiones negativas del otro. De este modo el cálculo de la correlación para un posicionamiento se puede realizar invirtiendo los valores de uno de los objetos:

$$C(P) = \int -M_\lambda^A \cdot \Lambda_P M_\lambda^B = \sum_x^X \sum_y^Y \sum_z^Z -M_\lambda^A[x,y,z] \cdot M_\lambda^B[\Lambda_P(x,y,z)] \qquad (2.2)$$

La maximización de esta función proporcionará la mejor complementariedad de la propiedad y el mejor contacto entre los objetos. Tanto (2.1) como (2.2) son expresiones generales que han de ser adaptadas a la naturaleza de cada

problema. En particular, la normalización de estas funciones variará según dicha naturaleza.

Por último, el criterio de ajuste puede depender de la correlación de distintas propiedades asociadas a los objetos. En tal caso, los valores de correlación deben ser combinados de forma adecuada mediante su suma ponderada:

$$C(P) = w_1 C_1(P) + w_2 C_2(P) + \ldots + w_n C_n(P) \qquad (2.3)$$

Donde los valores de ponderación w se establecen según la importancia relativa de cada propiedad.

2.2.1.3 Exploración

El conjunto de posibles posicionamientos entre objetos forma un espacio de búsqueda muy amplio. En el caso de ajuste de dos objetos, es necesario explorar un espacio 6-Dimensional compuesto por un espacio traslacional (posición espacial de un objeto con respecto a otro definida por tres valores cartesianos) y un espacio rotacional (rotación de un objeto con respecto al otro definida por tres valores angulares). La aproximación más simple a esta exploración (**Fig. 2.12**), consistente en un muestreo sistemático del espacio 6-Dimensional, resulta impracticable si se emplean intervalos de búsqueda relativamente pequeños. Suponiéndose un caso de ajuste interno en el que el intervalo rotacional se establezca a 6º, para cada traslación será necesario explorar más de 10^5 rotaciones. Si además los mapas tienen unas dimensiones de $100x100x100$ celdas o

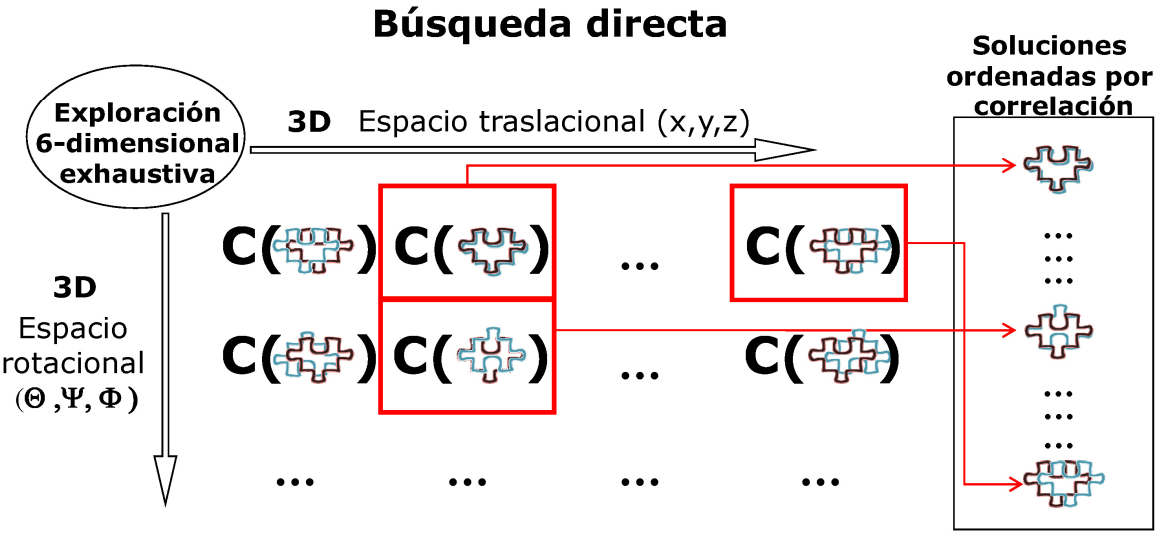

Fig.2.12- Esquema de búsqueda 6-Dimensional directa. La búsqueda 6-Dimensional explora de forma discreta el máximo número de posibles posiciones y orientaciones de un objeto con respecto a otro. Para cada una de estas combinaciones se calcula la correlación entre ambos objetos. En un último paso, las posibles soluciones son ordenadas en función de la correlación.

voxels, entonces el número de posibles translaciones puede ser de 10^6 (el centro de uno de los mapas es superpuesto sobre cada voxel del otro). En definitiva, un muestreo sistemático en este caso puede requerir explorar aproximadamente un centenar de miles de millones de posicionamientos de un mapa con respecto al otro, con el consiguiente coste computacional.

Para aumentar la eficiencia se han desarrollado metodologías que permiten acelerar la búsqueda en algunos de los grados de libertad que componen su espacio. La aproximación clásica empleada para acelerar esta búsqueda consiste en el cálculo de correlación en el espacio de frecuencias. Mediante el uso combinado del teorema de convolución y de técnicas de cálculo de la Transformada Rápida de Fourier (FFT, del inglés *Fast Fourier Transform*) es posible acelerar la búsqueda en el espacio traslacional, requiriéndose únicamente una exploración sistemática del espacio rotacional. De esta forma, para cada rotación, la correlación en el espacio de traslación es calculada mediante:

$$C(T) = \int M_\lambda^A \bullet M_\lambda^B = F^{-1}\left[\int F\left[M_\lambda^A[T]\right] \bullet F\left[M_\lambda^B[T]\right]\right] \qquad (2.4)$$

Este tipo de ajuste, denominado Ajuste Traslacional Rápido (FTM, del inglés *Fast Translational Matching*) [83], ha sido ya empleado con éxito en aplicaciones bioinformáticas de ajuste tanto interno [84] como externo [85]. De forma alternativa, también es posible acelerar la búsqueda sobre el espacio rotacional mediante la combinación de una representación adecuada del espacio rotacional y el empleo de armónicos esféricos [86-87]. Esta metodología, denominada ajuste rotacional rápido (FRM, del inglés *Fast Rotational Matching*) permite obtener una búsqueda mejor adaptada, y por tanto más rápida, a la naturaleza de los ajustes bioinformáticos que se describen a continuación. Por ello las aplicaciones de ajuste desarrolladas en esta tesis doctoral han sido realizadas a partir esta metodología.

2.2.1.4 Aplicaciones bioinformáticas

En el campo de la biología estructural los métodos de ajuste son de gran importancia debido a su utilidad en el modelado y predicción de estructuras de complejos macromoleculares. Si bien existen técnicas tales como la cristalografía de rayos X (XRC, del inglés *X-ray Crystallography*) [88] y la resonancia magnética (NMR, del inglés *Nuclear Magnetic Resonance*) [89] que permiten obtener una descripción de la estructura atómica de moléculas, el estudio de complejos macromoleculares mediante estas técnicas presenta dificultades prácticas debidas

al gran tamaño que éstos pueden presentar y a la dificultad que conlleva mantener su estabilidad en condiciones experimentales. Así, por ejemplo, de la gran cantidad de estructuras resueltas depositadas en el *Protein Data Bank* (PDB) [90], sólo una pequeña proporción corresponde a complejos macromoleculares.

Conocer la estructura tridimensional de complejos formados por diferentes proteínas resulta esencial para entender los procesos celulares tanto patológicos como naturales. En último término, estas estructuras tridimensionales pueden ser empleadas en aplicaciones prácticas como el diseño de fármacos. Así, no sólo resulta útil determinar qué proteínas interaccionan entre sí sino que también es necesario determinar cómo lo hacen.

Los métodos de ajuste pueden ser empleados en la predicción de la estructura de complejos macromoleculares en dos contextos distintos.

Ajuste Multi-resolución

Un primer método de ajuste se emplea para combinar la información a distinta resolución acerca de un complejo, de forma que las estructuras a nivel atómico de componentes del complejo son encajadas en mapas tridimensionales de menor resolución. En este caso, como criterio de ajuste se emplea la correlación entre las densidades electrónicas. Los mapas a baja resolución de un complejo se obtienen mediante Microscopía Electrónica (EM, del inglés *Electron Microscopy*) [91]. Esta técnica, si bien no permite obtener mapas con un grado de resolución lo suficientemente bajo como para determinar la estructura atómica, presenta menos restricciones en su utilización que las técnicas de mayor resolución como XRC o NMR. Mediante este método de ajuste interno la estructura del complejo puede ser construida como si de un puzle se tratara, encajando piezas a alta resolución dentro de un marco de baja resolución (ver **Fig. 2.13**). Este tipo de ajuste es conocido como ajuste multi-resolución (en inglés *Multiresolution Docking*) y ha sido desarrollado en diversas herramientas tales como COLORES [92] COAN [93], DOCKEM [94] o EMFIT [95].

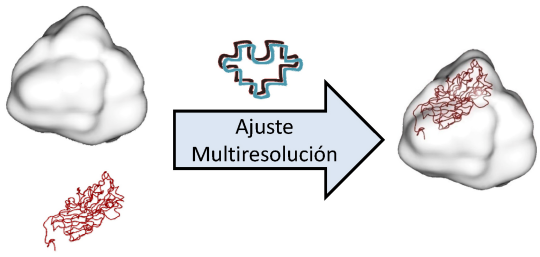

Fig.2.13- Ajuste multi-resolución interno. Una o varias estructuras atómicas se localizan dentro de un mapa a baja resolución.

Ajuste Proteína-Proteína

Un segundo escenario más complejo surge cuando no se tiene ninguna información acerca del

complejo macromolecular. En este caso se ha de predecir cómo interaccionan tridimensionalmente dos proteínas a partir de sus estructuras atómicas obtenidas de forma aislada. Esto se realiza mediante un ajuste de tipo externo (ver **Fig. 2.14**). Dado que las conformaciones con mayor estabilidad serán aquéllas con un menor grado de energía libre, el criterio de ajuste se orienta a lograr la mayor correlación entre términos energéticos complementarios. Debido a su importancia, este tipo de ajuste, denominado ajuste proteína-proteína (en inglés *Protein-Protein docking*) [96], ha sido afrontado mediante un amplio abanico de técnicas. Existen aproximaciones que tratan de predecir las superficies que presentan mayor posibilidad de estar presentes en la interacción [97-99] o determinar los sitios activos o residuos más funcionales [100] de los componentes. Otras alternativas emplean el conocimiento previo acerca de interacciones, adecuadamente almacenado en bases de datos tales como PQS [101] o DIP [102], para tratar de predecir nuevas interacciones [103-104]. También aspectos como la flexibilidad de los componentes, tanto de las cadenas laterales de los aminoácidos [105-106] como de la estructura principal [107-109], son estudiados.

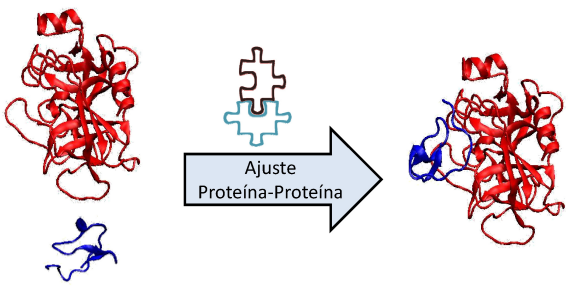

Fig.2.14- Ajuste proteína-proteína externo. Búsqueda de la mejor interacción entre dos estructuras atómicas.

Sin embargo, la mayoría de las aproximaciones creadas emplean estas técnicas para refinar predicciones obtenidas mediante una exploración exhaustiva del espacio de posicionamientos entre los componentes rígidos. Por ello, la adecuada realización de este ajuste rígido inicial resulta esencial. Este ajuste inicial ha sido usualmente realizado mediante el método FTM [84, 110-111] si bien también existen aplicaciones que emplean esféricos armónicos de forma similar a FRM [86].

Para la realización de ambos tipos de aplicaciones bioinformáticas se necesita encontrar un equilibrio entre fiabilidad, de forma que se obtengan estructuras de complejos correctas, y eficiencia, de modo que la búsqueda del mejor ajuste sea lo más rápida posible. En este sentido, el ajuste rotacional rápido (FRM) muestra características que lo hacen idóneo para realizar este tipo de ajuste. Así, siendo el ajuste entre proteínas muy sensible a variaciones rotacionales, resulta muy

ventajoso que la aceleración de la búsqueda sea realizada en el espacio rotacional, permitiendo realizar una exploración más detallada de éste. Por otro lado, en el ajuste entre proteínas es muy simple determinar limitaciones en el espacio traslacional a explorar, por lo que el muestreo sistemático de este espacio puede ser fácilmente reducido.

En la presente tesis doctoral se muestra el diseño y desarrollo de una novedosa metodología que, partiendo del método FRM, permite realizar ajustes de forma eficaz tanto para en el caso multi-resolución como para el caso proteína-proteína. Sin embargo, a pesar de la aceleración obtenida, la realización de estos ajustes sobre datos experimentales reales requiere un extenso tiempo de cómputo. Además, en muchos escenarios prácticos estos ajustes deberán ser repetidos múltiples veces con variaciones en el conjunto de datos de entrada. Por estos motivos, y dado que ambas aplicaciones pueden describirse como de alta productividad, su adaptación sobre sistemas Grid empleando el sistema cache propuesto es de gran utilidad.

3 Sistema cache de "extremo a extremo" sobre metaplanificación

Entre los múltiples servicios que deben ser proporcionados por un sistema Grid, una comunicación eficiente de datos resulta esencial. Debido a que un Grid es un sistema distribuido que conecta recursos localizados a grandes distancias, el empleo eficiente tanto de los recursos de conexión como de almacenamiento es de gran importancia para minimizar tiempos de latencia en la transferencia de datos. Aunque servicios existentes, como GridFTP y RFT en Globus, permiten la transmisión de datos entre recursos de un sistema Grid, no proporcionan políticas que permitan una organización eficiente de dicha transmisión.

Desde el punto de vista del tráfico de datos, un sistema Grid presenta características similares a las del sistema de comunicación a través de la red mundial WWW (del inglés *World Wide Web*). Aunque las características de la información que transita en ambos entornos no es exactamente igual (la comunicación es más intensa en WWW pero generalmente con menor carga de datos en los mensajes, el modelo de comunicación es diferente, etc.), los mecanismos empleados para mejorar el tránsito de información en uno pueden ser adaptados con el mismo fin en el otro. En este sentido, técnicas cache que

emplean mecanismos de almacenamiento, replicación y gestión de datos en repositorios distribuidos y temporales, han demostrado ser útiles en la reducción de latencias en el tráfico de datos a través de Internet [112]. Por tanto, la adecuación de estas técnicas a la naturaleza de los entornos Grid se presenta como una clara alternativa para mejorar la eficiencia en la transferencia de datos.

Un sistema cache en WWW es definido por diferentes características: localización del repositorio de almacenamiento (en los recursos de los clientes, en los recursos de un determinado servicio, en los recursos que actúan como canal de comunicación), el nivel de transparencia del sistema ante los usuarios, la capacidad de comunicación entre diferentes elementos cache localizados en distintos recursos, políticas de refresco de datos, políticas de consistencia, etc. Todas estas características, correctamente adaptadas, proporcionan mecanismos para crear sistemas cache adecuados para la computación Grid. Esta adaptación tendrá que emplear los diferentes servicios que los *middleware* Grid proporcionan para crear las funcionalidades cache, así como tener en cuenta las características de las aplicaciones a las que se quiera dar soporte.

Siguiendo estas consideraciones, la implantación de un sistema cache ha sido abordada desde diferentes puntos de vista. Las propuestas realizadas varían tanto en el diseño del sistema propuesto como en la funcionalidad que proporciona, pudiéndose emplear tanto nuevos módulos de *software* integrados en el *middleware* del Grid como nuevos elementos *hardware* introducidos en el sistema. La siguiente sección muestra los planteamientos más relevantes. Posteriormente se describen los principios que han guiado la creación del sistema cache propuesto en esta tesis doctoral. En posteriores secciones se detalla su diseño así como su adaptación a un metaplanificador concreto.

3.1 Aproximaciones a funcionalidad cache sobre sistemas Grid

3.1.1 Aproximaciones software

Mediante la interacción con las funcionalidades proporcionadas tanto por el *middleware* del sistema Grid como por los servicios locales de los recursos, es posible desarrollar *software* que, convenientemente integrado en el *middleware*, permite proporcionar funcionalidades cache:

- Entre los diferentes servicios proporcionados en las versiones iniciales de GT se encontraba GASS (del inglés *Global Access to Secondary Storage*).

Este servicio, basado en el protocolo HTTP, proporcionaba a aplicaciones funcionalidad para el acceso remoto a ficheros sobre un sistema Grid [113]. Entre las características que incorporaba GASS, se encontraba estrategias de movimiento de datos que mantenían almacenamientos cache sobre los distintos recursos, permitiendo de este modo reducir la transferencia de datos real sobre el sistema Grid. De esta forma, GASS mantenía un sistema cache multi-usuario y distribuido en el sistema Grid. Sin embargo, este sistema cache presentaba un grave problema de escalabilidad debido a la necesidad mantener réplicas de datos. Este problema estaba derivado de un diseño del servicio demasiado similar al empleado en la comunicación HTTP. Debido a ello, además de otros inconvenientes, GASS fue eliminado de la versión 4 de GT. Sin embargo, las prestaciones cache que el servicio GASS proporcionaba son muy adecuadas para el tipo de aplicaciones bioinformáticas cuyo rendimiento se busca mejorar en esta tesis doctoral. Por ello, el sistema cache diseñado proporcionará una funcionalidad que tratará de suplir la falta de este servicio.

- Yonny *et al*. [114] han propuesto una infraestructura básica para la gestión conjunta y colaborativa de diferentes mecanismos cache en un entorno Grid. Este sistema cooperativo gestiona el intercambio tanto de datos como de metadatos entre diferentes caches. Estas caches se encuentran implantadas en los diferentes recursos Grid mediante el uso tanto de servicios Grid como de soporte local. Al ser una infraestructura orientada al soporte de aplicaciones científicas, características como la coherencia o consistencia en la duplicación de datos no son consideradas debido a la naturaleza, únicamente de lectura, de los datos tratados.

3.1.2 Aproximaciones hardware

La funcionalidad cache también puede ser proporcionada por elementos dedicados. Esta aproximación requiere, además de modificaciones del *software*, la introducción de nuevos recursos *hardware* en el sistema Grid:

- Es posible establecer un sistema cache remoto y compartido por todos los recursos del entorno Grid [115]. Así, esta es una solución de orientación *hardware*, siendo introducidos elementos físicos en el entorno Grid que actúan como servidores de la funcionalidad cache. En este sentido, el

concepto de cache empleado aquí es el de almacenamiento temporal de rápido acceso. Los elementos introducidos para proporcionar capacidades cache se denominan sistemas de almacenamiento distribuido y paralelo o DPSS (del inglés *Distributed-Parallel Storage System*). De forma esquemática, un DPSS está compuesto por un conjunto de computadores de bajo coste, cada uno de los cuales contiene varios discos de acceso paralelo. Los datos almacenados en cache son distribuidos sobre estos computadores, permitiéndose el acceso paralelo y concurrente. Los computadores mantienen una porción de los datos *cacheados* accesible al entorno Grid mientras que los discos paralelos actúan como almacenamiento terciario guardando el conjunto de información *cacheada*. Tanto las fuentes como los consumidores de información acceden al DPSS a través del entorno Grid para la introducción, modificación y lectura de datos. En resumen, este sistema está orientado a permitir un acceso rápido a grandes cantidades de datos producidos por aplicaciones científicas con flujo de datos de alta velocidad. En este sentido, aspectos como la política de replicación no son tenidos en cuenta al no requerirse replicas de datos.

- Elementos llamados Gestores de Recursos de Almacenamiento (SRM, *Storage Resource Manager*) han sido introducidos como componentes Grid para dar soporte al almacenamiento de amplios conjuntos de datos [116]. Con el fin de proporcionar un acceso a datos eficiente, han sido propuestas políticas de reemplazo cache sobre SRM adaptadas a la transferencia de datos en Grids [117].

3.1.3 Funcionalidades similares

También existen aproximaciones que, sin llegar a ser específicamente soluciones que proporcionen funcionalidad cache en entornos Grid, emplean conceptos relativos a técnicas de cache para proporcionar diferentes funcionalidades Grid. En [118] se describe un servicio de replicación de datos de alto nivel. Básicamente, servicios establecidos en distintos sitios del entorno Grid almacenan información replicada, transfiriéndola a través del entorno Grid cuando es requerida. Si bien la política de replicación empleada en este caso es similar a la usada en servicios cache, el objetivo de este servicio es la rápida difusión de información a través de entornos Grid, sin tener como objetivo la reducción del tráfico en dichos entornos.

3.2 Características del sistema cache

Con el fin de proporcionar un diseño del sistema cache adecuado, este debe ser abordado teniendo en cuenta tanto las características del sistema sobre el que va a ser implantado como el tipo de aplicaciones a las que se desea proporcionar servicio. En este sentido los dos aspectos fundamentales a tener en cuenta son:

- ***Implantación sobre sistemas Grid***: El sistema cache deberá de respetar las características de la computación Grid. Principalmente, tendrá que gestionar de manera adecuada la autonomía de los recursos, así como la naturaleza inherentemente distribuida de este tipo de sistemas.
- ***Soporte a aplicaciones de alta productividad***: El principal objetivo es la mejora del rendimiento de aplicaciones de alta productividad que realicen un uso intensivo de la misma información. La consecución de este objetivo guiará la forma en la que la información deberá ser gestionada en el sistema cache.

A partir de estos aspectos se derivan las siguientes características:

1) Implantación sobre metaplanificación

Todos los mecanismos cache que se han indicado en la sección anterior presentan un importante aspecto que repercute de forma sustancial en su implantación: requieren la introducción de elementos, que pueden ser tanto *hardware* como *software*, en el entorno Grid. Debido a la naturaleza distribuida de los sistemas Grid, la introducción de estos elementos supone un proceso complejo. Tareas como la diseminación de *software* a través de los recursos y administraciones de un Grid, establecimiento de nuevos recursos *hardware* y modificación de servicios o protocolos ya establecidos pueden suponer un coste muy elevado. Además, estas acciones intrusivas deberán de contar con el respaldo y colaboración de los diferentes agentes involucrados en el uso y administración del entorno Grid. Aunque estas complejas tareas no son inabordables y forman parte del proceso de creación y evolución de la computación Grid, el alto coste que conllevan puede hacer inviable la incorporación de un sistema cache.

Para evitar la necesidad de una compleja implantación intrusiva, el sistema cache se diseñará aprovechando las funcionalidades proporcionadas por los sistemas de metaplanificación. Un metaplanificador, gracias a su comunicación

con los diferentes servicios proporcionados por el *middleware*, posee las capacidades necesarias para crear y gestionar zonas de almacenamiento temporal en diferentes recursos. De esta forma, los cambios necesarios se limitan a una modificación en el diseño del metaplanificador, que siendo una herramienta de alto nivel (véase **Fig. 2.2**), no afecta al funcionamiento de otros servicios de la arquitectura Grid. Además, habitualmente un metaplanificador es un módulo *software* que únicamente se ejecuta en un recurso (el del usuario que realiza las tareas o un recurso que actúa como servidor). De este modo, la adaptación del metaplanificador para proporcionar funcionalidad cache es una solución simple y poco costosa, ya que no necesita diseminar modificaciones en el entorno Grid.

2) Esquema "extremo a extremo"

Debido a que los diferentes recursos que conforman un sistema Grid se encuentran distribuidos tanto geográfica como administrativamente, el sistema cache también tendrá que ser distribuido. Para cada elemento de computación del Grid, por tanto, se establecerá un repositorio de almacenamiento de la información cache. Sin embargo, la gestión y mantenimiento de dichos repositorios, tal y como se ha indicado anteriormente, serán realizados por un metaplanificador desde el nodo de acceso del usuario. De esta manera, el mecanismo mediante el que la funcionalidad cache es proporcionada sigue un esquema de "extremo a extremo": los directorios o repositorios cache se mantienen en recursos empleados para la ejecución de tareas mientras que su gestión se controla desde el metaplanificador en el recurso del usuario (ver **Fig. 3.1**).

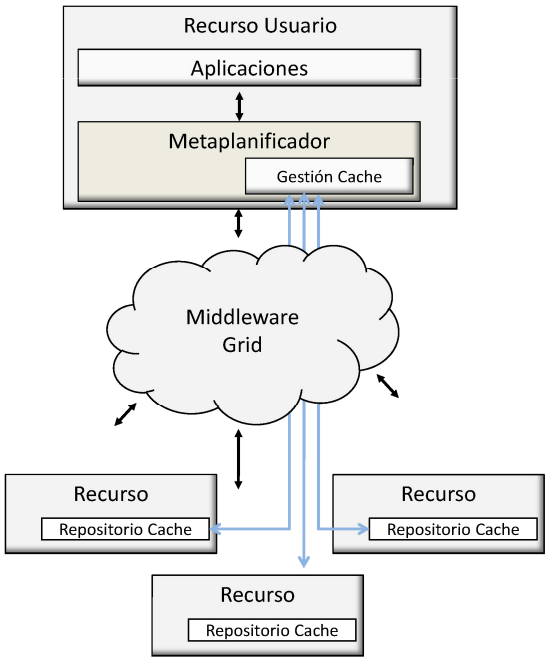

Fig. 3.1- Esquema del sistema cache "extremo a extremo". La gestión del sistema cache es controlada desde el recurso de usuario por parte del metaplanificador. Los diferentes repositorios son mantenidos en los recursos de cómputo.

3) Determinación unívoca de la información

Para adecuarse a la naturaleza distribuida del sistema Grid así como para evitar transferencias de datos innecesarias, se hace necesario un

etiquetado de la información contenida en los repositorios cache que evite redundancias. Por tanto, los ficheros que sean almacenados en estos repositorios deberán de ser nombrados con un identificador único.

4) Gestión de repositorios delegada

Debido a que los repositorios cache son gestionados bajo la administración autónoma de cada recurso, algunos de los aspectos de su gestión por parte del metaplanificador pueden ser innecesarios. Así, no es posible determinar con fiabilidad ni la capacidad máxima ni el tiempo durante el que los datos introducidos en dichos repositorios serán accesibles. Con respecto a la capacidad máxima, esta dependerá de las cuotas de almacenamiento que la administración local proporcione a los usuarios externos, dato que no siempre será conocido. En cuanto al tiempo de vida de los datos en cache, es esperable que las administraciones locales no permitan un mantenimiento ilimitado de los datos de los usuarios externos, forzando la eliminación de dichos datos tras un periodo de tiempo establecido. Por tanto, la determinación de sustituciones de estos datos no es prioritaria siendo muy complicado llegar a saturar el almacenamiento disponible. Por otro lado, al estar el sistema cache enfocado a mejorar el rendimiento de aplicaciones paralelas con tareas compartiendo información común, es esperable que las tareas requieran únicamente datos instanciados recientemente por tareas de la misma aplicación, no empleándose datos de otras aplicaciones anteriores. De este modo, más que una política de reemplazo, se hacen necesarios mecanismos para eliminar los datos de una determinada aplicación cuando estos se consideren innecesarios.

5) Privacidad de la información

Si bien usualmente la compartición de datos almacenados en cache por parte de diferentes usuarios puede resultar beneficiosa, la propia naturaleza de las aplicaciones objetivo hace a esta compartición poco útil. La principal ventaja que se pretende alcanzar con el sistema cache propuesto es la de permitir que tareas pertenecientes a una misma aplicación paralela puedan compartir datos transmitidos una sola vez. De este modo, no es esperable que tareas de distintos usuarios puedan compartir información. Además, desde la perspectiva de la seguridad, el mantenimiento de repositorios de acceso común para diferentes usuarios puede permitir el acceso por parte de éstos a datos confidenciales de otros usuarios. Por último, la viabilidad de establecer repositorios comunes es compleja debido a la naturaleza no intrusiva del sistema cache propuesto. Debido

a que toda la gestión cache se realiza mediante la funcionalidad del metaplanificador, se restringe en gran medida los derechos de acceso a recursos remotos. La creación de repositorios cache está limitada a zonas de almacenamiento proporcionadas por las administraciones locales para cada usuario. De este modo, resulta imposible establecer un repositorio de acceso común para distintos usuarios. Aunque mecanismos de replicación de datos podrían paliar esta limitación, la complejidad que añadiría al sistema cache puede producir problemas de escalabilidad similares a los mostrados por el servicio GASS de Globus (véase sección **3.1.1**).

3.3 Diseño del sistema cache

Las características indicadas en el apartado anterior establecen las bases para el diseño del sistema cache propuesto en esta tesis doctoral [119]. Así, el establecimiento de los componentes indicados en **Fig. 3.1** permite gestionar el sistema cache desde el metaplanificador manteniendo el esquema "extremo a extremo":

- ***Gestor de cache***: El metaplanificador tendrá que ser modificado para proporcionar las funcionalidades de gestión del sistema cache. Para ello, el metaplanificador almacenará información acerca de los datos transmitidos, tratará de forma adecuada dicha información y establecerá mecanismos de transferencia adecuados a la naturaleza de los recursos empleados. Los elementos que tendrán que incluirse o modificarse son:
  - Almacén de información *cacheada*: El metaplanificador mantendrá información acerca de la distribución de la información en los distintos recursos.
  - Gestión de transferencia: Las operaciones de transferencia de datos a través del sistema Grid estarán precedidas de operaciones para determinar su conveniencia. Asimismo, la creación y transferencia de datos a los repositorios cache modificarán el modo en que estas operaciones son realizadas.
  - Rutina de control remoto: Para realizar operaciones sobre los recursos, un metaplanificador puede emplear la ejecución remota de rutinas preestablecidas. Estas rutinas serán modificadas para introducir la lógica de utilización de los datos almacenados en repositorios cache.

- ***Repositorios***: Serán simples directorios establecidos sobre el espacio de almacenamiento proporcionado a los usuarios Grid. Estos repositorios serán accedidos por el metaplanificador tanto para instanciar ficheros como para emplear dichos ficheros en la ejecución de una tarea.

Para realizar un determinado tarea sobre un sistema Grid, un usuario realiza una petición al agente de metaplanifición, indicando las características y requerimientos de ésta y proporcionando los ficheros ejecutables y de entrada necesarios. El metaplanificador por su parte, teniendo en cuenta los requerimientos indicados, determina qué recurso o recursos realizarán la ejecución. En cada recurso, el metaplanificador gestionará un repositorio cache en el que se mantendrán copias de ficheros empleados por las diferentes tareas. De esta forma, cuando una tarea requiera un determinado fichero en un recurso, el metaplanificador determinará si una copia de este fichero está presente en el repositorio del recurso o por el contrario el fichero debe ser transferido desde el recurso del usuario. Si el fichero ha de ser transferido, se instanciará en el repositorio cache del recurso. Posteriormente se establecerán enlaces a aquellos ficheros del repositorio necesarios para la ejecución de la tarea.

De este modo, al realizarse la ejecución de una tarea en un determinado nodo de cómputo, las acciones que un metaplanificador debe llevar a cabo referentes a la gestión de los repositorios cache son:

```
Crear conjunto MD5 = ficheros marcados como cacheables y etiquetados con MD5.
IF repositorio en recurso elegido no existe:
    THEN Crear repositorio
FOR EACH f IN MD5
    IF f no existe en repositorio:
        THEN transferir f al repositorio
Enlazar MD5 desde repositorio a directorio de trabajo de tarea
IF MD5 transferidos correctamente:
    THEN Actualizar información cache.
```

Mediante el etiquetado por MD5 (*Message-Digest algorithm 5*) [120] se obtiene una identificación unívoca de los ficheros. Por otro lado, la privacidad de los repositorios queda garantizada al establecerse éstos sobre las cuotas de espacio asignadas a cada usuario.

3.3.1 Políticas Cache

El establecimiento del sistema cache requiere la toma de decisiones en algunos de los aspectos de su gestión. Principalmente, son dos los aspectos que deben ser estudiados. Por un lado, en un sistema cache es muy importante establecer una política de reemplazo de datos *cacheables* adecuada. Por otro lado, debido a la naturaleza distribuida del sistema, la política de transferencia jugará un importante papel en las funcionalidades proporcionadas por el sistema cache.

3.3.1.1 Política de reemplazo

Generalmente el establecimiento de un repositorio cache conlleva la necesidad de establecer un mecanismo para sustituir datos antiguos por datos de nuevo uso. Las políticas que se han estudiado para realizar esta sustitución [121] están orientadas a realizar un óptimo uso del espacio disponible para el almacenamiento cache, manteniendo aquellos datos que más pueden ser reutilizados.

Sin embargo, tal y como se ha indicado en la sección **3.2**, la autonomía de los recursos en un Grid limita la utilidad de este tipo de políticas. Así, debido a que no es posible determinar la capacidad límite de un repositorio y a que el tiempo de vida de éste y de sus datos estará controlado por la política local, no se ha establecido una política de reemplazo en el sistema cache.

3.3.1.2 Políticas de transferencia

Un aspecto fundamental, que juega un papel muy importante en la forma en que se realizan las acciones de gestión del sistema cache, es la accesibilidad al almacenamiento de los nodos de cómputo por parte de un metaplanificador. En función de esta accesibilidad, un metaplanificador empleará dos esquemas de transferencia de datos diferentes para interactuar con los recursos. Un esquema de los dos posibles escenarios de almacenamiento se muestra en **Fig. 3.2**.

Esquema de transferencia directo

Generalmente los conjuntos de nodos de cómputo de un sitio o recurso no son visibles directamente desde el entorno Grid, produciéndose su acceso a través de un nodo de enlace. El nodo de enlace actúa como interfaz entre el conjunto de nodos computacionales de un recurso y el sistema Grid, recibiendo las peticiones de cómputo y determinando el nodo o nodos del recurso que serán empleados. Por tanto, toda la comunicación establecida por un metaplanificador con un determinado recurso se establece a través de su nodo de enlace. Si este nodo

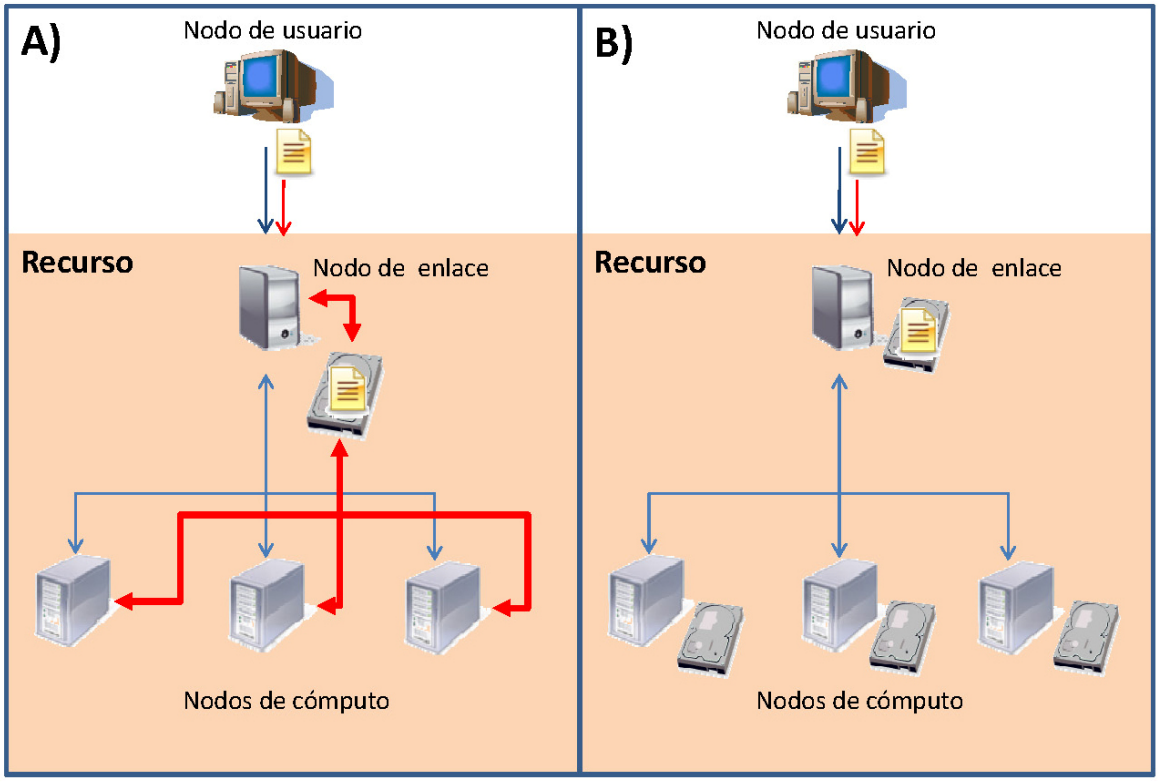

Fig.3.2- Diferentes escenarios de almacenamiento en recursos Grid. Cuando el almacenamiento secundario es compartido por todos los nodos de un recurso (A), la transferencia de un fichero al nodo de enlace implica que todos los nodos de cómputo tienen acceso a él. En este caso se empleará un esquema de transferencia directo. Cuando el almacenamiento no es compartido (B), la realización de la transferencia al nodo de enlace no permite el acceso de los nodos de cómputo. En este caso, el esquema de transferencia empleado es inverso.

comparte su almacenamiento secundario con todo el conjunto de nodos de cómputo, entonces una transferencia de ficheros a este nodo permite que todo el conjunto tenga acceso a ellos. En este sentido, a efectos de metaplanificación, un recurso Grid puede considerarse como una unidad computacional en sí mismo. En tal caso, el metaplanificador puede establecer un esquema de transferencia directo, comenzando las transferencias de ficheros desde el nodo de usuario al nodo de enlace, ya que, sea cual sea el nodo de cómputo empleado posteriormente, tendrá acceso a la información transmitida.

Esquema de transferencia inverso

Un escenario más complejo surge cuando los diferentes nodos localizados en un mismo recurso no presentan compartición de almacenamiento secundario. En estos casos, la transferencia desde el nodo del usuario al nodo enlace no garantiza que el nodo de cómputo posteriormente empleado tenga acceso a la información transmitida. Por tanto, en este tipo de recursos la transferencia debe ser inicializada desde el nodo de cómputo una vez éste ha sido elegido, pudiéndose así conocer tanto el destino (nodo de cómputo empleado) como el

origen (nodo de usuario) de la comunicación. En este escenario, el metaplanificador debe de proporcionar un esquema de transferencia invertido para interactuar con este tipo de recursos, en el cual la transferencia de datos se inicializa desde el destino. En tal situación, un recurso no puede considerarse una unidad computacional individual, ya que el hecho de transmitir información a un nodo del recurso no implica que el resto tenga acceso a dicha información.

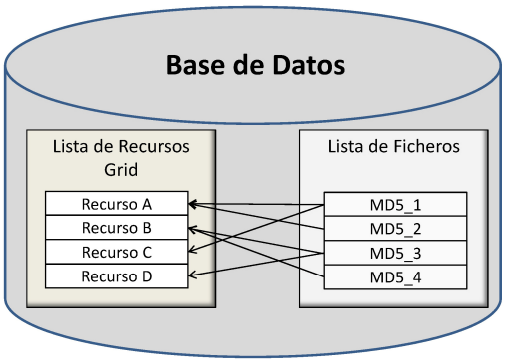

Fig.3.3- Base de datos del sistema cache mantenida por el metaplanificador. La base de datos mantiene una lista de recursos accedidos por el metaplanificador y de ficheros que han sido introducidos en los repositorios de los recursos. Las flechas indican la existencia de un determinado fichero en un recurso.

La existencia de distintos esquemas de transferencia obliga a mantener el sistema cache de distinto modo en función del esquema empleado. Por esta razón, se ha diseñado la gestión del sistema cache empleando dos políticas de transferencia diferentes: la **Política Centralizada** se emplea cuando el esquema de transferencia es directo y el almacenamiento en todos los recursos es completamente compartido; por otro lado la **Política Remota** se utiliza cuando el esquema de transferencia es inverso, pudiéndose emplear tanto si los recursos muestran almacenamiento compartido como si no.

3.3.1.2.1 Política centralizada

Al garantizarse que la transferencia de ficheros a un nodo permite su futuro acceso para cualquier otro nodo de cómputo del mismo recurso, el conjunto de nodos de un recurso puede ser tratado como un único nodo de cómputo, con un repositorio cache común.

De este modo, para gestionar adecuadamente los repositorios cache de los diferentes recursos, el metaplanificador mantiene una base de datos local que contiene información sobre las instancias de ficheros que pueden encontrarse en cada uno de los repositorios del sistema Grid. **Fig. 3.3** muestra esquemáticamente la estructura de la base de datos. El metaplanificador ha de interactuar con esta base de datos modificándola conforme se realicen las diferentes operaciones sobre los recursos del sistema Grid. Dicha interacción viene definida por las siguientes operaciones:

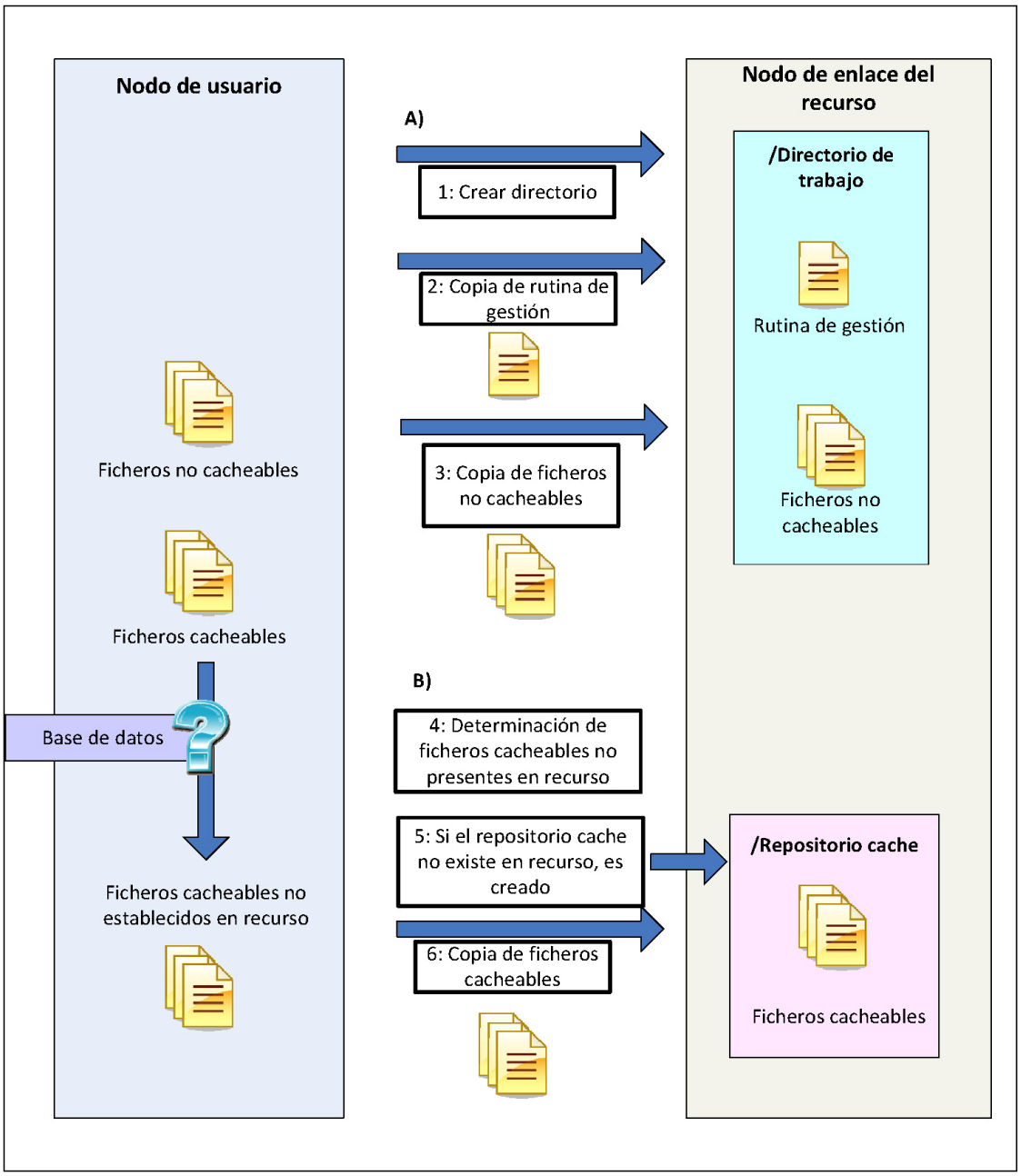

Fig.3.4- Establecimiento de una tarea en un recurso con la política centralizada del sistema cache. Las operaciones básicas del metaplanificador (**A**) son la creación de un directorio de trabajo (1) y la transferencia tanto de la rutina de gestión de la tarea (2) como de aquellos ficheros de entrada que no se consideren *cacheables* (3). Para la gestión de la cache (**B**), el metaplanificador además debe de chequear en la base de datos la existencia de ficheros *cacheables* en el recurso elegido (4) y transferir aquellos que no se encuentren (6), creando un repositorio cache sobre el recurso si no existe uno previo (5).

- ***Introducción de entrada***: Introduce una nueva instancia de un fichero en un recurso.
- ***Borrado de entrada***: Se elimina la instancia de un fichero en un recurso.
- ***Chequeo de entrada***: Determina si una instancia de un fichero se encuentra disponible en un recurso.

- ***Recuperación de todas las entradas***: Devuelve la colección completa de entradas en la base de datos.

Con el fin de evitar inconsistencias, todas estas operaciones son realizadas empleando identificadores MD5 de los ficheros. Cada una de estas operaciones se emplea durante el ciclo de ejecución de una tarea. De esta forma, durante la fase preliminar de la ejecución de una tarea y tras seleccionar un recurso, el metaplanificador realiza los pasos descritos en **Fig. 3.4**. Así, además de establecer un directorio de trabajo y transferir directamente una rutina de control y los ficheros de entrada no *cacheables*, el metaplanificador deberá realizar nuevas acciones orientadas a gestionar la cache. El metaplanificador consultará la base de datos para determinar la existencia de ficheros *cacheables* en el repositorio del recurso elegido. Únicamente si un fichero no se encuentra será transferido. Asimismo, el repositorio será creado sobre el recurso si no existe previamente.

Posteriormente, la rutina de control de la tarea se ejecuta sobre uno de los nodos de cómputo del recurso. Previamente al lanzamiento de la tarea, esta

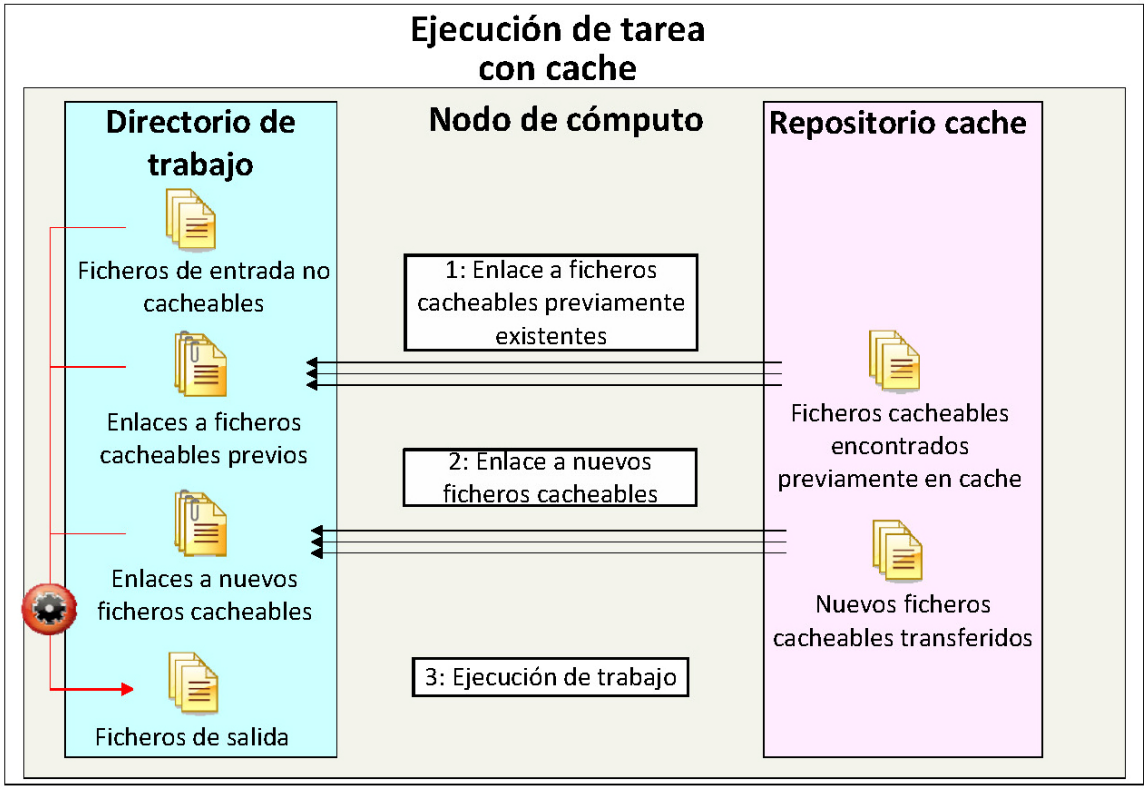

Fig.3.5- Acciones realizadas por la rutina de control para gestionar el repositorio en la política centralizada del sistema cache. Antes de la ejecución de la tarea (3), la rutina enlaza tanto los ficheros necesarios del repositorio cache que se encontraban previamente en él (1) como aquellos que han sido transferidos desde el nodo del usuario en el paso anterior (2).

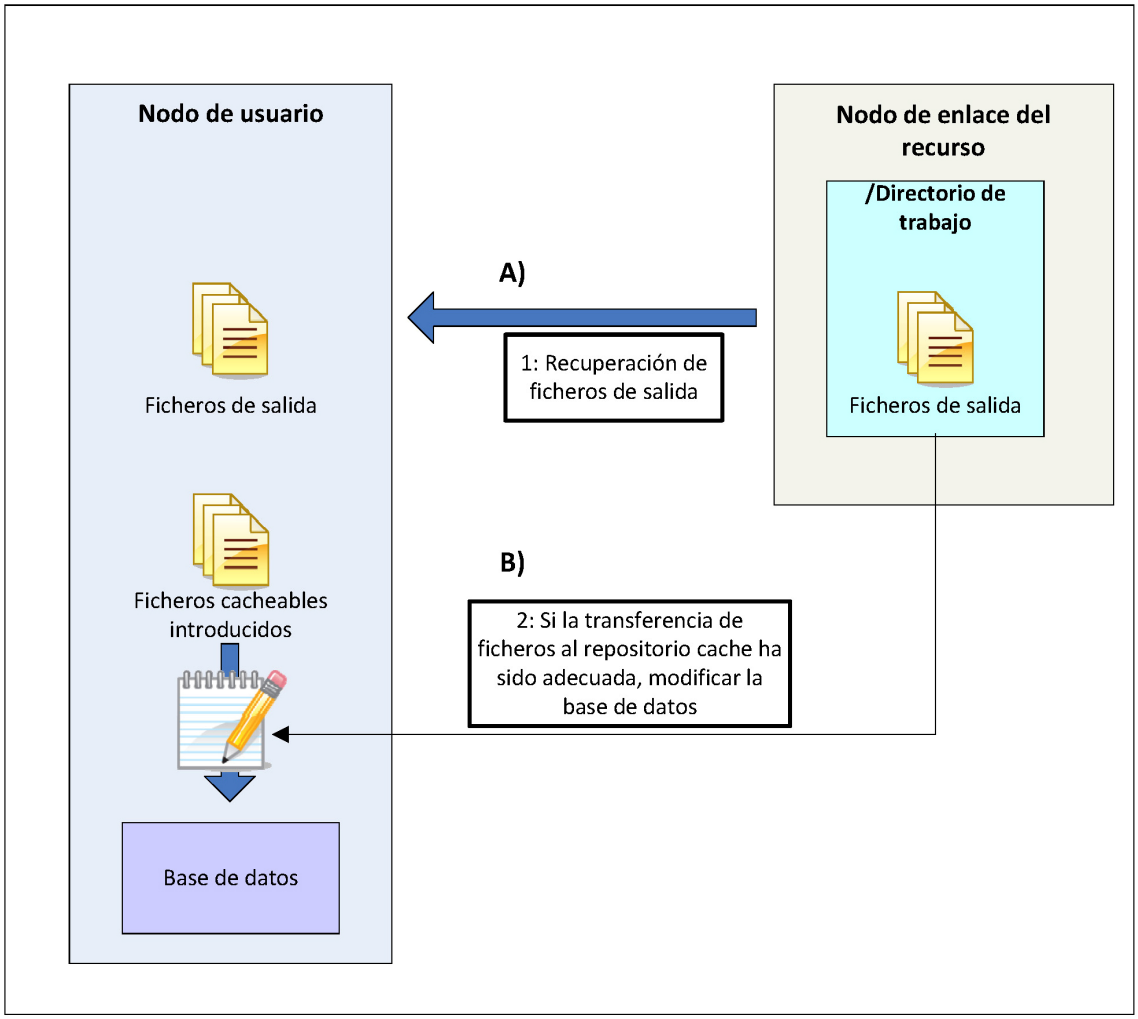

Fig.3.6- Acciones del metaplanificador tras la ejecución correcta de la tarea con la política centralizada del sistema cache. Además de las acciones básicas de gestión de la tarea (**A)** como la transferencia al nodo de usuario de los resultados (1), la gestión del sistema cache (**B**) requiere una actualización de la base de datos indicando los nuevos ficheros presentes en el repositorio del recurso (2).

rutina ha de realizar acciones de enlazamiento, sobre el directorio donde la tarea se ejecuta, tanto de aquellos ficheros que han sido encontrados previamente en el repositorio como de aquellos que han tenido que ser transferidos (ver **Fig. 3.5**). Tras la consecución de estos pasos, la tarea puede ser ejecutada.

Por último, tras detectar la finalización de la tarea y el correcto alojamiento de los nuevos ficheros en el repositorio del recurso, el metaplanificador, además de recuperar los resultados, actualizará la base de datos indicando las nuevas instancias de ficheros. Este paso se realiza sólo tras obtener confirmación de la correcta transferencia de los ficheros (**Fig. 3.6**).

Además de estas operaciones durante el ciclo de ejecución de una tarea, el metaplanificador deberá de realizar comprobaciones periódicas de la consistencia de la información contenida en la base de datos. Esto es necesario debido a la falta de control sobre las políticas de almacenamiento de los recursos indicada

anteriormente. Por esta razón, no es posible determinar el tiempo de vida de un repositorio en un determinado recurso, pudiendo la administración local mantenerlo de forma indefinida, borrarlo tras un periodo de tiempo o incluso borrarlo tras la ejecución de la tarea que lo estableció (en este caso extremo, el sistema cache resultaría inútil sobre dicho recurso). Por tanto, el metaplanificador debe, a determinados intervalos, consultar las instancias de la base de datos y, para cada una de ellas, realizar una comprobación empleando servicios Grid para determinar la permanencia de los ficheros en los recursos. Si esta permanencia no puede determinarse, entonces su instancia es eliminada de la base de datos. En caso contrario la instancia es actualizada para mantener constancia del último acceso al fichero (ver **Fig. 3.7**).

El intervalo de actualización de la base de datos debe de ser lo suficientemente breve como para garantizar la confiabilidad en la información contenida en la base de datos. En cualquier caso, en el improbable caso en el que una tarea sea asignada a un recurso donde el repositorio cache haya sido borrado sin que la base de datos haya sido actualizada, la realización de la tarea produciría un error por falta de datos para la ejecución, lo que obligaría al metaplanificador a realizar de nuevo la tarea (bien en el mismo recurso, bien en otro distinto). Un intervalo apropiado para la actualización de la base de datos evitará una repetición de este

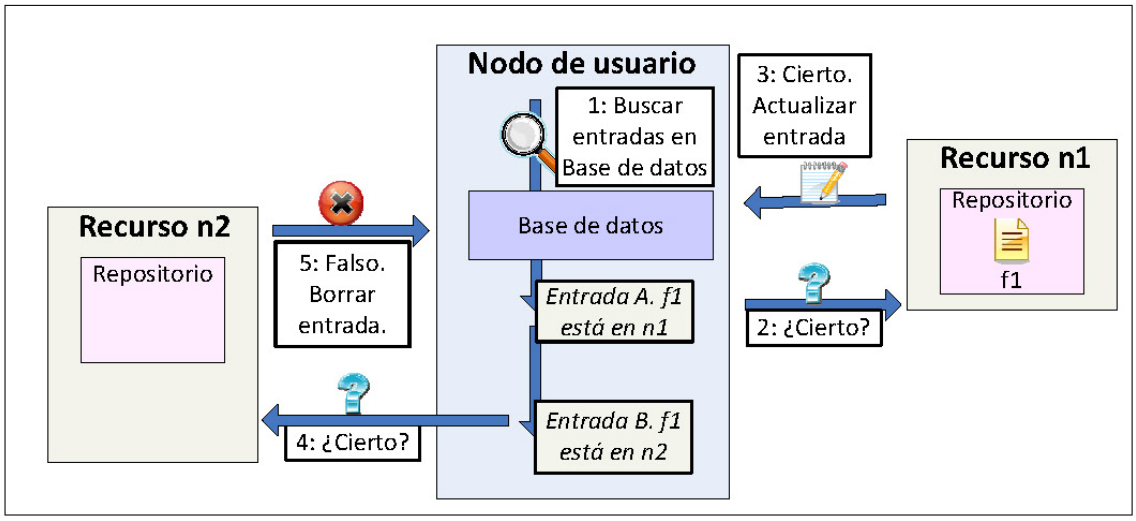

Fig.3.7- Actualización de base de datos por parte del metaplanificador en la política centralizada del sistema cache. El metaplanificador chequea todas las instancias introducidas en la base de datos (1). Mediante servicios *middleware* Grid el metaplanificador determina si una instancia es cierta (2). En caso de serlo se actualiza la instancia en la base de datos (3). En caso de ser falsa (4), la instancia es borrada de la base de datos (5).

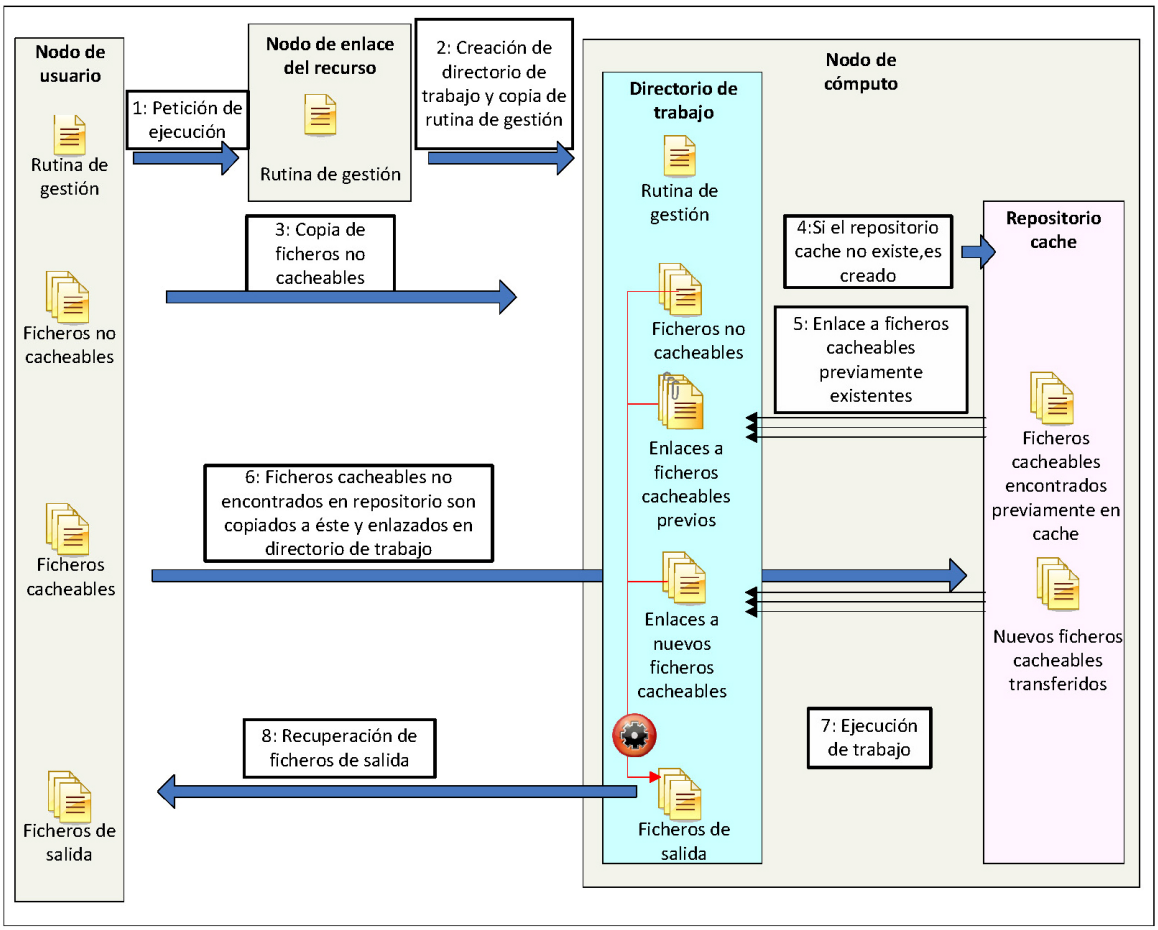

Fig.3.8- Esquema de ejecución de una tarea con la política remota del sistema cache. El metaplanificador realiza la petición de realización de un trabajo al nodo de enlace (1) de un recurso. A través de este nodo de enlace es seleccionado un nodo de cómputo donde el directorio de trabajo es creado y la rutina de control es transferida (2). Las siguientes acciones son realizadas por la rutina de control: Los ficheros no *cacheables* necesarios para la tarea son transferidos (3), el repositorio cache es creado si no existe (4), los ficheros *cacheables* necesarios previamente existentes en el repositorio son enlazados (5) mientras que aquellos no encontrados son transferidos al repositorio desde el nodo de usuario y enlazados posteriormente (6), la tarea es ejecutada (7) y finalmente los resultados son transferidos al nodo de usuario (8).

error. Si bien esta repetición de una tarea repercute negativamente en la productividad del sistema, la escasa probabilidad de que este tipo de situaciones se produzcan reduce su impacto, resultando irrelevante frente a las mejoras proporcionadas por el sistema cache.

3.3.1.2.2 Política remota

Esta política se emplea cuando el conjunto de nodos de cómputo de un recurso no comparte almacenamiento secundario y, por tanto, el esquema de transferencia es inverso. En este caso, la transferencia de un fichero a un recurso no garantiza su localización posterior en este, ya que puede ser empleado un nodo de cómputo diferente. Por tanto, el recurso no puede ser tratado como una unidad computacional y el mantenimiento de una base de datos local, almacenando instancias de ficheros copiadas en recursos, no es útil. Por ello, en

este escenario todas las operaciones de manejo de los repositorios deben ser realizadas desde el propio nodo mediante una rutina de control remoto sin el uso de una base de datos.

En la fase preliminar, el metaplanificador realiza las operaciones habituales para el establecimiento del directorio de trabajo, transfiriendo la rutina de control. La transferencia de ficheros, tanto *cacheables* como no, se realiza por la rutina de control cuando ésta es ejecutada por el nodo de cómputo seleccionado. De esta forma, la rutina crea un repositorio cache si éste no existía previamente. En caso contrario, comprueba la existencia de cada fichero *cacheable* en el repositorio. Si un fichero *cacheable* no se encuentra en el repositorio se transfiere, mediante servicios Grid, desde el nodo de usuario al repositorio cache; en caso contrario el fichero es enlazado desde el repositorio al directorio de trabajo. Igualmente, la rutina, tras las transferencias, realiza el enlazamiento de los nuevos ficheros *cacheables* copiados. Por último, también es la rutina de control la que, desde el nodo de cómputo, realizará las transferencias de resultados al nodo de usuario (ver **Fig. 3.8**).

3.3.1.2.3 Comparación entre políticas

La aplicación de ambas políticas supone un coste similar. Únicamente la política centralizada requiere una mayor carga en el tránsito de información a través del sistema Grid debido a la necesidad de realizar comprobaciones periódicas de la consistencia de la base de datos. Sin embargo, este incremento en el tránsito es despreciable ya que únicamente implica peticiones simples de comprobación de acceso a ficheros, por lo que la carga de datos asociada es mínima.

Desde el punto de vista de prestaciones, la política centralizada, a través de la base de datos local, proporciona al metaplanificador información acerca de la distribución de datos sobre el entorno Grid. De esta forma, el metaplanificador puede adaptar sus políticas de planificación para emplear dicha información, pudiendo así realizar asignaciones más eficientes de tareas. Conociendo la localización de ficheros *cacheables*, el planificador puede dar prioridad al uso de recursos que ya dispongan de ellos. La política remota, por el contrario, no proporciona ningún mecanismo mediante el cual el metaplanificador pueda determinar la localización de ficheros. Desde este punto de vista, la política remota puede también describirse como "ciega" y proporciona menores prestaciones que la centralizada.

Sin embargo, es la política remota la que ofrece una implantación más general, pudiéndose emplear en entornos Grid formados tanto por recursos con almacenamiento compartido como por recursos con almacenamiento independiente. Por el contrario, la política centralizada, aplicada sobre un esquema de transferencia directo, requiere que todos los recursos del entorno tengan almacenamiento secundario compartido por todos sus nodos. Por tanto, si bien la política centralizada proporciona una mayor prestación, la política remota es más versátil.

3.4 Adaptación del sistema cache sobre el metaplanificador GridWay

El diseño del sistema cache está orientado a su fácil implantación sobre cualquier metaplanificador. Para ello, se han de modificar los distintos módulos que lo componen para incluir las operaciones necesarias. Así mismo, en función de la política de transferencia empleada, los cambios a realizar serán de distinta índole.

Con el fin mostrar una adaptación del sistema cache, en esta tesis doctoral se ha empleado GridWay. GridWay es un metaplanificador desarrollado para interactuar con servicios proporcionados por Globus (Véase sección **2.1.4.2**). GridWay presenta las características adecuadas para permitir una fácil adaptación del sistema cache propuesto, siendo un componente software que únicamente ha de ser instalado sobre el nodo de usuario, de forma que su modificación no requiere una difusión de cambios a través del entorno Grid. Además, el diseño de GridWay ya ha sido adaptado a diferentes escenarios, proporcionando esquemas de trasferencia directo e inverso en función de las características de los entornos Grid a emplear. GridWay proporciona diferentes implementaciones tanto del gestor de transferencias como de la rutina de control remoto de tarea (véase sección **2.1.4.2.1**) para cada uno de estos escenarios. Básicamente, en el caso de que los recursos del entorno muestren almacenamiento compartido, el gestor de transferencia (que forma parte del agente de envíos ejecutado en el nodo de usuario) se encarga de hacer las transferencias de datos pertinentes a través del nodo de enlace. En caso contrario las transferencias son delegadas a la rutina de control remoto. La existencia de estas dos implementaciones facilita la adaptación de ambas políticas cache. Dicha adaptación requiere diferentes modificaciones en los módulos de GridWay para realizar las acciones descritas para cada política.

3.4.1 Política Centralizada en GridWay

A continuación se describen las modificaciones necesarias realizadas en los módulos de GridWay pertinentes:

- ***Gestor de peticiones***: La estructura de comunicación ha sido modificada con el fin de permitir al usuario determinar qué ficheros deben ser almacenados en los repositorios cache. La comunicación entre el usuario y el gestor de peticiones se realiza a través de un fichero descriptor. Este fichero contiene diferentes campos en los que el usuario indica el ejecutable utilizado, los ficheros de entrada necesarios, los requerimientos de los recursos a emplear, etc. Este módulo ha sido modificado para incluir dos variables:

  a) Permite al usuario indicar ficheros de entrada *cacheables.*

  b) Permite indicar si el ejecutable debe ser mantenido en cache.

  En cualquier caso, también es posible forzar la consideración de todos los ficheros de entrada como *cacheables*. Por último, también ha sido modificado el tratamiento de la información de entrada, incluyendo la realización del cálculo de los identificadores MD5 con los que son etiquetados los ficheros al ser introducidos en los repositorios cache.

- ***Gestor de transferencia***: El gestor de transferencia ha sido modificado para forzar la comprobación acerca de la conveniencia en la transmisión de datos. De este modo, cuando se recibe una petición para transferir un conjunto de ficheros a un determinado recurso, se comprueba mediante una consulta a la base de datos la existencia de instancias en el recurso de todos aquellos ficheros que hayan sido etiquetados con una marca MD5 (y por tanto son *cacheables*). Aquellos para los que la respuesta es negativa son transferidos. Para ello, a la etiqueta de cada fichero (formado por su identificador MD5) se le añaden identificadores tanto del usuario como de la tarea correspondiente. Esta etiqueta compuesta, y exclusiva para cada fichero y tarea, se emplea como nombre de destino para el fichero sobre el repositorio cache. Mediante este etiquetado unívoco para cada tarea se evitan colisiones en la transferencia concurrente de ficheros. Esta transferencia concurrente puede producirse al ser aceptadas al mismo tiempo varias tareas (empleando los mismos ficheros) por un mismo

recurso. De esta forma, varias tareas comenzarán a transmitir los mismos ficheros concurrentemente, ya que ninguna de ellas habrá tenido tiempo de modificar la base de datos. Si se emplea el mismo nombre en la transferencia de todas las tareas, la concurrencia en la transmisión puede provocar errores en la ejecución de alguna de éstas, ya que cuando una tarea comience su ejecución sus ficheros de entrada pueden ser modificados por otra tarea que continúe transfiriéndolos. Mediante el etiquetado unívoco se evita esta concurrencia y ficheros ya transmitidos no son modificados. Sin embargo, esta solución requiere volver a etiquetar correctamente los ficheros una vez estos han sido completamente transmitidos. Esta función será realizada por la rutina de control remoto. El esquema del mecanismo de transferencia se encuentra representado en **Fig. 3.9**.

- ***Rutina de control remoto***: La rutina empleada en esta política se corresponde a la asociada con el esquema de transferencia directo. Las modificaciones realizadas en esta rutina han sido orientadas al correcto acceso a la información *cacheable*. Previamente a la ejecución de la tarea sobre el nodo de cómputo elegido, esta rutina comprueba si han sido transmitidos al repositorio cache ficheros con el etiquetado unívoco de la tarea. En caso de que así sea, y si los ficheros no han sido previamente establecidos por otra tarea concurrente, la rutina vuelve a etiquetar estos ficheros con su identificador MD5 (eliminando los identificadores de usuario y tarea). En caso de que los ficheros ya hayan sido establecidos, la rutina únicamente borra los ficheros con etiqueta unívoca. Por último, se realizan las operaciones de enlace tanto de los ficheros previamente existentes en el repositorio como de los nuevos ficheros recién transferidos.
- ***Gestor de Ejecución***: Este módulo ha sido modificado para actualizar la base de datos. Tras el establecimiento de nuevos ficheros en el repositorio cache de un recurso, nuevas instancias deben de ser insertadas en la base de datos. Con el fin de mantener la consistencia de la base de datos es esencial asegurar el correcto establecimiento remoto de los ficheros. La garantía de un establecimiento correcto sólo puede ser obtenida tras la correcta realización de la fase de preparación del entorno de la tarea (fase *prolog*. Véase sección **2.1.4.2.2**). Las operaciones de copia de ficheros por

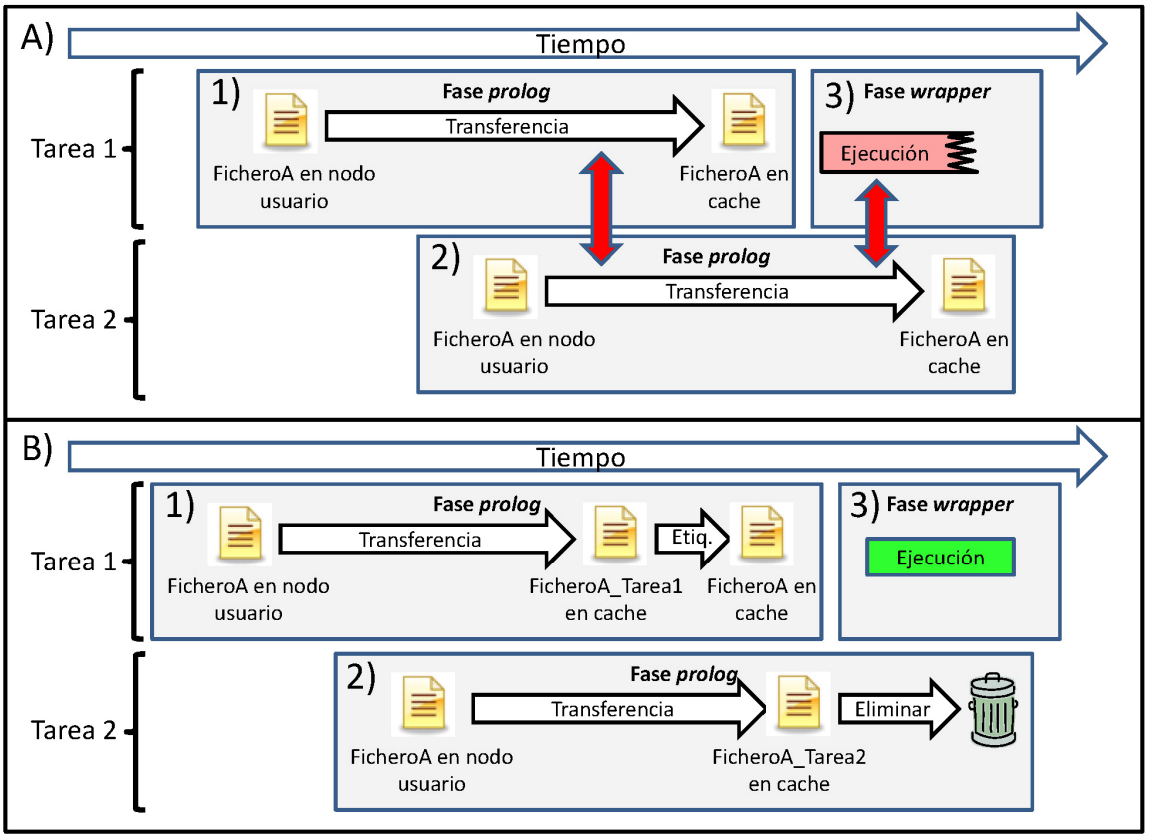

Fig.3.9- Concurrencia en la transferencia de un fichero (**FicheroA**) a un mismo recurso por parte de dos tareas. En **(A)** se muestra los posibles errores producidos por el uso del mismo identificador del fichero en la transferencia. Si la Tarea 2 comienza su realización en un escaso margen de tiempo con respecto al comienzo de la Tarea 1, no detectará la transferencia realizada por esta última **(1)** e iniciará su propia transferencia **(2)**. Al emplear ambas tareas la misma etiqueta para el fichero de destino (**FicheroA**) se producirá una concurrencia en su escritura. Del mismo modo, si la Tarea 2 continúa su transferencia coincidiendo con la ejecución de la Tarea 1 **(3)** se producirán modificaciones en el fichero que producirán la incorrecta realización de dicha ejecución. Estos errores son evitados empleando un etiquetado diferente para cada tarea **(B)**. En este caso, las fases de *Prolog* no colisionan al emplear nombres de destino distintos (**FicheroA_Tarea1** y **FicheroA_Tarea2**). En **1)**, tras realizar la transferencia el ficheros se etiqueta con su nombre correcto. En **2)**, como se detecta que el fichero correctamente etiquetado existe, se elimina el fichero transmitido. De esta forma, tanto la ejecución de la Tarea 1 **(3)** como de la Tarea 2 emplearán la instancia del fichero transmitida por la Tarea 1.

parte del gestor de transferencias son realizadas durante esta fase, de modo que a su finalización los ficheros estarán establecidos en el repositorio cache. Por tanto, tras la finalización de la fase *prolog*, el gestor de ejecución interacciona con la base de datos introduciendo las instancias asociadas a los ficheros copiados. Es importante recordar que el gestor de transferencias transmite los ficheros nombrados con una etiqueta unívoca para cada tarea. De esta forma, al finalizar la fase *prolog*, los ficheros no se encontrarán en el repositorio cache con la etiqueta adecuada. Sin embargo, como se ha indicado, la rutina de control remoto etiqueta correctamente los ficheros antes de comenzar la ejecución. Como la tarea será ejecutada con anterioridad a posteriores tareas asignadas al recurso, puede garantizarse que dichas tareas encontrarán los ficheros correctamente etiquetados.

- ***Monitor de Rendimiento***: La modificación de este modulo es necesaria para garantizar la consistencia de la base de datos. De este modo, junto con las acciones de comprobación de rendimiento de las tareas, el monitor de rendimiento también realizará periódicamente comprobaciones de la consistencia de la base de datos. Para ello el monitor lee todas las instancias introducidas en la base de datos y comprueba su veracidad a través de la interacción con el servicio GridFTP de Globus. En caso de que un fichero haya sido borrado de un repositorio, la correspondiente instancia es eliminada de la base de datos.

3.4.2 Política Remota en GridWay

Debido a que con esta política la transferencia de ficheros se controla desde el nodo de cómputo remoto, y a que no se mantiene una base de datos local, la mayoría de las modificaciones de los módulos anteriores no son necesarias, realizándose las acciones sobre la rutina de control remoto:

- ***Gestor de peticiones***: Las modificaciones en este módulo son idénticas a las indicadas en la política anterior. De esta forma, el gestor de peticiones se encarga de etiquetar con identificadores MD5 aquellos ficheros indicados por el usuario.
- ***Rutina de control remoto***: La rutina de control empleada está asociada al esquema de transferencia inverso. Debido a que las transferencias de ficheros deben de ser comenzadas desde el nodo de cómputo, las acciones de gestión de los repositorios recaen en la rutina de control, y por tanto nuevas modificaciones son añadidas. Además de las acciones indicadas para la anterior política, la rutina de control debe comprobar la existencia de ficheros *cacheables* en el repositorio cache del nodo, enlazándolos desde el directorio de trabajo si existen o transfiriéndolos (y enlazándolos posteriormente) desde el nodo usuario en caso contrario. Al igual que la anterior política, los ficheros son transferidos etiquetados unívocamente para cada tarea. Una vez han sido transferidos, y si no existe ya una copia de ellos con la etiqueta correcta, los ficheros son reetiquetados. Por último, la rutina realiza la transferencia al nodo de usuario de los ficheros de salida resultantes.

4 Aplicaciones bioinformáticas de ajuste

En este capítulo se describe el novedoso desarrollo de dos aplicaciones para realizar los tipos de ajuste bioinformático descritos en la introducción. Para ello, en una primera sección se detallan los principios del algoritmo de búsqueda, conocido como Ajuste Rotacional Rápido, que se ha empleado. En esta sección se muestran las alternativas de uso de este método y se justifica su utilización. En una segunda sección se describe el desarrollo de las diferentes aplicaciones basadas en este método. Por último, se detalla la forma en que estas aplicaciones han sido paralelizadas con el fin de poder ser empleadas sobre un sistema Grid.

4.1 Estrategia de exploración. Ajuste Rotacional Rápido.

Como brevemente se ha descrito en la sección **2.2.1.3**, el método FTM permite una aceleración sobre las dimensiones cartesianas del espacio de búsqueda mediante el teorema de convolución y el empleo de la FFT. Así, en este método, la búsqueda de ajustes entre dos objetos se realiza mediante un muestreo sistemático del espacio de rotación y, para cada rotación explorada, los valores de correlación para todo el espacio de traslación son calculados en el espacio de frecuencias (ver **Fig. 4.1**).

Mientras el espacio cartesiano que describe la traslación entre los objetos muestra un comportamiento más uniforme y presumiblemente menos complejo, el espacio angular de rotación puede mostrar variaciones más heterogéneas, dificultando la detección de máximos y mostrando muchas degeneraciones de una

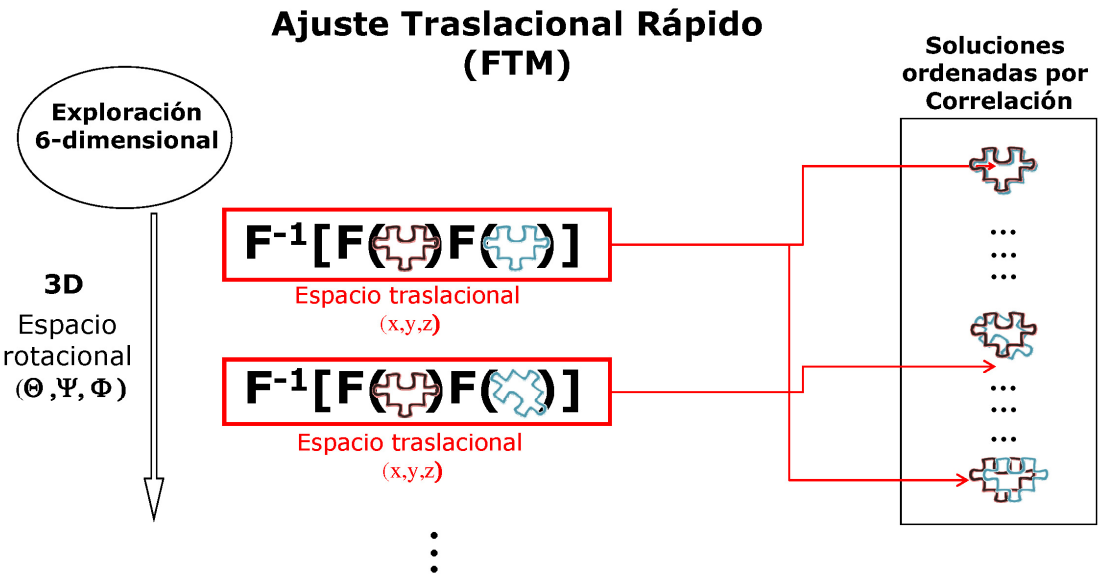

Fig.4.1- Esquema de búsqueda 6-Dimensional con aceleración en el espacio traslacional (FTM). El espacio rotacional se muestrea de forma sistemática en intervalos discretos. Para cada rotación aplicada a un objeto tridimensional, las correlaciones en el espacio de traslacional se calculan mediante la FFT. Las traslaciones que obtienen mejor correlación son almacenadas en la lista final de soluciones.

misma solución. Por ello, es más conveniente realizar un muestreo más fino del espacio rotacional, por lo que una aceleración en este espacio de búsqueda resulta más eficiente. Por tanto, resulta más adecuado aplicar una aceleración similar a la anterior sobre dimensiones angulares y muestrear sistemáticamente las dimensiones cartesianas. Para llevar a cabo este Ajuste Rotacional Rápido (FRM), además de emplear el cálculo de la FFT, se realiza la aceleración de la búsqueda en el espacio rotacional gracias a una expresión de las matrices tridimensionales en esféricos armónicos y a una adecuada representación del espacio rotacional SO(3). Por otro lado, un muestreo sistemático sobre dimensiones cartesianas es más fácil de limitar que sobre dimensiones rotacionales ya que es posible establecer restricciones en las distancias entre los objetos, evitando posicionamientos que *a priori* se consideran no válidos.

El desarrollo más simple de FRM permite acelerar la exploración de las tres dimensiones angulares que conforman el espacio rotacional de la búsqueda. Sin embargo, el espacio 6-Dimensional que debe ser explorado también puede ser descrito mediante la combinación de cinco dimensiones angulares y una sola dimensión cartesiana, de forma que el método FRM puede ser empleado para acelerar la búsqueda de cinco grados de libertad en lugar de sólo tres. Las bases metodológicas de ambas adaptaciones se describen en las siguientes secciones.

4.1.1 Ajuste rotacional rápido sobre tres grados de libertad (FRM3)

En principio, el ajuste limitado al espacio rotacional entre dos objetos puede ser realizado calculando la correlación mediante un muestreo sistemático de los tres grados de libertad angulares que describen todas las posibles rotaciones de uno de los objetos frente al otro. Una alternativa mucho más rápida descubierta por Crowter permite calcular la correlación entre los objetos acelerando la búsqueda en el espacio de rotación mediante el uso de FFT y expresando los objetos mediante esféricos armónicos [122]. Si bien la aproximación de Crowter sólo aceleraba dos de los grados de libertad en el espacio de rotación, teniendo que muestrearse aún el tercero, el método ha sido ampliado recientemente para cubrir los tres grados de libertad [87] de la forma que se describe a continuación y se ilustra en **Fig. 4.2**.

Tal y como se indicó en la sección **2.2.1.2** y sin tener en cuenta desplazamientos traslacionales, para una determinada rotación el cálculo de la correlación entre f y g, siendo ambas propiedades de los objetos

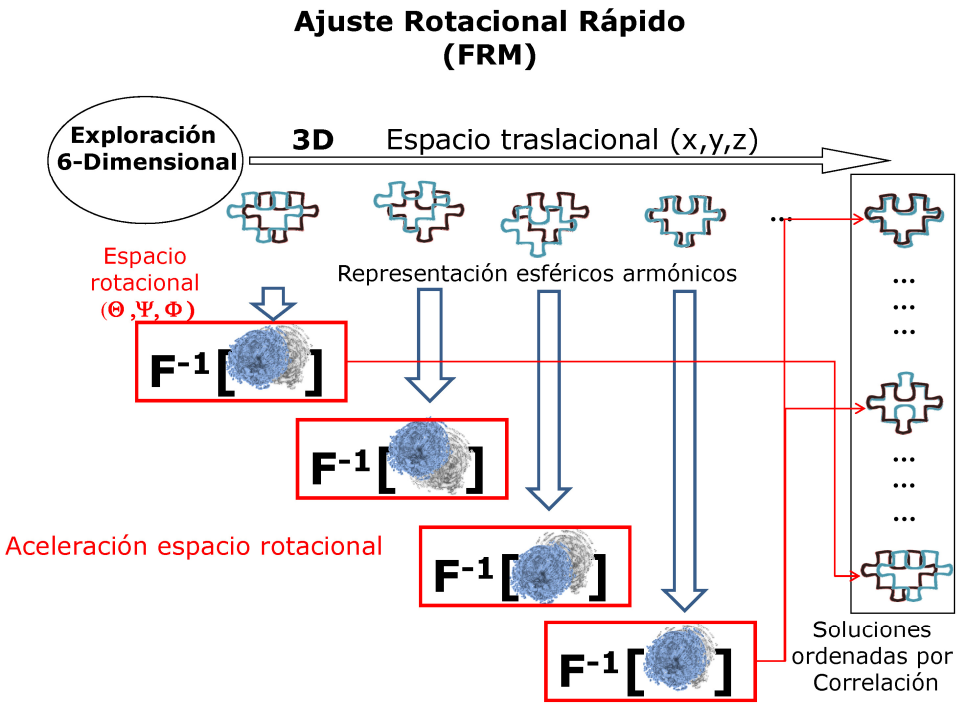

Fig.4.2- Esquema de búsqueda 6-Dimensional acelerando el espacio rotacional (FRM). El espacio traslacional se muestrea de forma sistemática en intervalos discretos definidos *a priori*. Para cada traslación el cálculo de correlaciones en el espacio rotacional es acelerado mediante una adecuada representación del espacio rotacional y el empleo de esféricos armónicos. Ver texto para más detalles. Las rotaciones que obtienen mejor correlación son almacenadas en la lista de soluciones.

tridimensionales a ajustar, se define por la integral:

$$C(R) = \int f \cdot \Lambda_R g \tag{4.1}$$

Siendo Λ_R un operador de rotación y R una rotación definida por una terna de ángulos de Euler $\{\varphi, \phi, \psi\}$.

Si un punto $p \in \mathbb{R}^3$ es definido por sus coordenadas polares, respecto a un centro de coordenadas establecido, como $p = \{r, \beta, \lambda\}$ donde $r = |p|$ es el radio y $\{\beta, \lambda\}$ son respectivamente la longitud y latitud, entonces f y g pueden ser expresadas en una expansión radial de sumas finitas de esféricos armónicos, los cuales forman una base ortogonal en la esfera unidad:

$$\begin{aligned} f(r,\beta,\lambda) &= \sum_{l=0}^{B-1} \sum_{m=-l}^{l} \hat{f}_{lm}(r) Y_{lm}(\beta,\lambda) \\ g(r,\beta,\lambda) &= \sum_{l=0}^{B-1} \sum_{m=-l}^{l} \hat{g}_{lm}(r) Y_{lm}(\beta,\lambda) \end{aligned} \tag{4.2}$$

Siendo Y_{lm} las funciones armónicas esféricas, donde $l \geq 0$ y $-l \leq m \leq l$. Estas funciones pueden ser descritas en términos de funciones de Legrende asociadas P_l^m [123]:

$$\begin{aligned} Y_{lm} &: \left[S^2 = \left\{ \{\beta, \lambda\} \in \mathbb{R}^3 \,|\, |\beta, \lambda| = 1 \right\} \right] \to \mathbb{C} \\ Y_{lm}(\beta,\lambda) &= (-1)^m \left[\frac{(2l+1)(l-m)!}{4\pi/(l+m)!} \right]^{1/2} P_l^m(\cos\beta) e^{im\lambda} \end{aligned} \tag{4.3}$$

$\hat{f}_{lm}(r)$ y $\hat{g}_{lm}(r)$ son los coeficientes asociados a los esféricos armónicos que permiten aproximar la combinación de éstos a los valores de f y g en la capa de radio *r*. Así, esta representación describe las matrices como una serie de capas concéntricas separadas por un intervalo establecido y estando cada capa expandida en una base de esféricos armónicos. *B* es el orden de la representación esférica y limita el número de esféricos armónicos empleado para aproximar cada capa. El valor *B* también determina el número de puntos muestreados sobre dicha capa, de tal manera que el intervalo de muestreo viene dado por $360/2B$. Así, por ejemplo, para *B*=16 se realizarán muestreos cada 11.25°. Con este ancho de banda, y teniendo en cuenta que el espacio de rotación es completamente cubierto mediante tres ángulos de Euler de los cuales dos varían en el intervalo [0:2π] y el tercero lo hace en [0:π] [124], el número total de rotaciones exploradas es de $32 \cdot 32 \cdot 16 \approx 16000$ rotaciones. En realidad el número de rotaciones

efectivo es algo menor debido a la existencia de degeneraciones de los ángulos de Euler. Dicho de otro modo, una misma rotación puede ser descrita con diferentes combinaciones de ángulos de Euler.

Si se define el operador de rotación Λ_R empleado en (4.1) para una rotación *R* en el grupo rotacional tridimensional SO(3) [125] como:

$$(\Lambda_R g)(r,\beta,\lambda) \coloneqq g\left[R^{-1}(r,\beta,\lambda)\right] = g\left[rR^{-1}(\beta,\lambda)\right] \quad \forall\{r,\beta,\lambda\} \in \mathbb{R}^3 \tag{4.4}$$

La aplicación de una rotación a un esférico armónico puede transformarse en una combinación lineal de otros esféricos de igual grado:

$$Y_{lm}[R^{-1}(\beta,\lambda)] = \sum_n D^l_{nm}(R) Y_{\ln}(\beta,\lambda) \tag{4.5}$$

Por tanto, la aplicación de una rotación a una matriz cumple:

$$\begin{aligned}(\Lambda_R g)(r,\beta,\lambda) = g\left[R^{-1}(r,\beta,\lambda)\right] = g\left[rR^{-1}(\beta,\lambda)\right] = \\ \sum_{l,m} \hat{g}_{lm}(r) Y_{lm}[R^{-1}(\beta,\lambda)] = \sum_{l,m,n} \hat{g}_{lm}(r) D^l_{nm}(R) Y_{\ln}(\beta,\lambda)\end{aligned} \tag{4.6}$$

Los valores D^l_{nm} son valores constantes de las matrices de la representación irreducible de SO(3). Básicamente, las matrices D proporcionan un cambio del sistema de coordenadas para la representación sobre la base de esféricos armónicos [126]. Estos valores de las matrices D pueden ser calculados eficientemente a partir de la serie de funciones d^l_{mn} mediante un proceso recursivo [127]. Así, si se define una rotación como una combinación de tres ángulos de Euler $R = \{\varphi,\theta,\psi\}$, los valores para D^l_{nm} pueden calcularse como:

$$D^l_{mn}(\varphi,\theta,\psi) = e^{-im\varphi} d^l_{mn}(\theta) e^{-in\psi} \tag{4.7}$$

A partir de las ecuaciones (4.2) y (4.6) la correlación entre un objeto y una versión rotada del otro se puede definir como:

$$c(R) = \sum_{r,\beta,\lambda} f(r,\beta,\lambda) \cdot \overline{\Lambda_R g(r,\beta,\lambda)} = \sum_{ll'mm'n} \overline{D^{l'}_{nm'}(R)} \sum_{r,\beta,\lambda} \hat{f}_{lm}(r) \overline{\hat{g}_{l'm'}(r)} Y_{\ln}(\beta,\lambda) \overline{Y_{l'n}(\beta,\lambda)} \tag{4.8}$$

El complejo conjugado sobre el segundo objeto es introducido con el fin de evitar la necesidad de un cambio de variables posterior. Teniendo en cuenta la ortogonalidad de los esféricos armónicos, la integral queda reducida a:

$$\sum_{r,\beta,\lambda} \hat{f}_{lm}(r) \overline{\hat{g}_{l'm'}(r)} Y_{lm}(\beta,\lambda) \overline{Y_{l'n}(\beta,\lambda)} = \delta_{ll'} \delta_{mn} I^l_{mm'} \tag{4.9}$$

Donde:

$$I^l_{mm'} = \sum_r \hat{f}_{lm}(r) \overline{\hat{g}_{lm'}(r)} r^2 \tag{4.10}$$

Pudiéndose expresar la correlación como:

$$c(R) = \sum_{r,\beta,\lambda} f(r,\beta,\lambda) \cdot \overline{\Lambda_R g(r,\beta,\lambda)} = \sum_{lmm'} \overline{D^{l'}_{mm'}(R)} I^l_{mm'} \tag{4.11}$$

Siendo δ la función delta de Kronecker definida como 1 para subíndices iguales y 0 para subíndices distintos. Con el fin de sólo depender de coeficientes d^l_{mn} calculados sobre un único ángulo, la rotación $R=\{\varphi,\phi,\psi\}$ se puede factorizar en dos rotaciones con el siguiente cambio de variables:

$$\begin{aligned} &\xi = \varphi - \pi/2, \quad \eta = \pi - \phi, \quad \omega = \psi - \pi/2 \\ &R = \{\varphi,\phi,\psi\} = R_1 R_2 = \{\xi, \pi/2, 0\}\{\eta, \pi/2, \omega\} \end{aligned} \tag{4.12}$$

Y teniendo en cuenta que

$$D^l_{nm}(R_1 \cdot R_2) = \sum_{h=m}^{n} D^l_{nh}(R_1) D^l_{hm}(R_2) \tag{4.13}$$

Se obtiene:

$$D^l_{nm}(R) = \sum_h d^l_{nh}(\pi/2) d^l_{hm}(\pi/2) e^{-i(n\xi + h\eta + m\omega)} \tag{4.14}$$

Y sustituyendo (4.14) en (4.11):

$$c(R) := C(\varphi,\theta,\psi) = \sum_{lmm'} \left[\sum_h \left[d^l_{mh}(\pi/2) d^l_{hm'}(\pi/2) e^{i(m(\varphi-\pi/2)+h(\pi-\theta)+m'(\psi-\pi/2))} \right] I^l_{mm'} \right] \tag{4.15}$$

Por último, calculando esta ecuación mediante su transformada de Fourier y posterior transformada inversa se obtiene:

$$c(R) = F^{-1}\left[\hat{C}(m,h,m') \right] = F^{-1}\left[\sum_l d^l_{mh}(\pi/2) d^l_{hm'}(\pi/2) I^l_{mm'} \right] \tag{4.16}$$

Esta transformada inversa, calculada eficientemente mediante FFT, proporciona la correlación entre f y g en el espacio rotacional definido por R=$\{\varphi,\theta,\psi\}$. Esta formulación proporciona una serie de ventajas que permiten acelerar la búsqueda:

- Mientras que el cálculo de la correlación en el espacio real es de orden cuadrático $O(M^2)$, siendo M el número de posibles orientaciones exploradas, mediante el uso de la FRM el número de operaciones realizadas escala según $O(M \log^2 M)$ [83, 128].
- La integral definida en (4.10) presenta simetrías [128-129]. Debido a ello, sólo es necesario calcular sobre el espacio complejo de frecuencias una

matriz con una de sus dimensiones reducida a la mitad, proporcionando la transformada inversa una matriz de dimensiones correctas. Por ello (4.16) puede calcularse sólo para $-B < m,h < B$ y $0 <= m' < B$. Aún así, la matriz de correlaciones obtenida para el espacio rotacional presenta simetrías que permiten reducir el cálculo.

- Las sumatorias de los coeficientes d^l_{mn} pueden ser pre-calculadas, siendo únicamente necesario calcularlas para el ángulo $\pi/2$. Además, este cálculo puede realizarse eficientemente mediante un proceso recursivo. Por tanto, para un *B* fijo estos coeficientes pueden ser reutilizadas para el cálculo de correlación rotacional para todas las posibles translaciones.
- Si en la exploración del espacio traslacional de búsqueda se mantiene fijo g, sus coeficientes armónicos $\hat{g}_{lm}$ pueden ser pre-calculados empleando como origen su propio centro. De esta forma, en cada traslación explorada, el cálculo de la correlación rotacional (4.16) sólo requiere calcular los coeficientes armónicos $\hat{f}_{lm}$. Para ello, debe considerarse como origen de sus expansiones en esféricos armónicos el centro del objeto g, de forma que para cada traslación la expansión del objeto f variará su origen en función de la posición de g.
- Debido a las propiedades de la transformada de Fourier se puede demostrar que:

$$w_A F^{-1}(A) + w_B F^{-1}(B) = F^{-1}(w_A A + w_B B) \qquad (4.17)$$

  Esto permite calcular la combinación de múltiples correlaciones para dos objetos (representando diferentes propiedades y ponderadas por diferentes valores) mediante una sola transformada inversa.
- Por otro lado, es importante resaltar que siempre se emplea el mismo número de muestreos rotacionales en todas las capas concéntricas, de modo que esta representación esférica permite un mayor nivel de detalle para las capas más interiores (y con menor área) mientras que la resolución es menor para capas más alejadas (**Fig. 4.3**). Este aspecto ha de ser considerado con detenimiento, ya que puede acarrear perdida de fiabilidad en el ajuste en casos en que la correlación de las capas más

externas sea importante. Por tanto, al afrontar la resolución de un problema de ajuste, es necesario favorecer que las zonas de interacción más importantes entre los dos objetos estén próximas al origen de las expansiones armónicas.

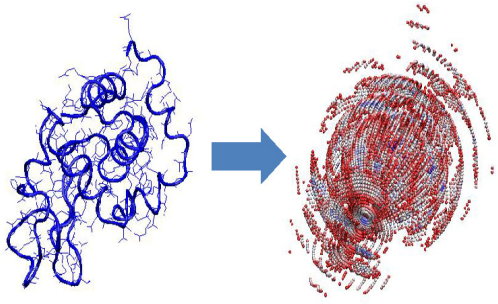

Fig.4.3- Muestreo esférico y radial de una propiedad de un objeto tridimensional. La figura a la derecha representa el muestreo radial de la densidad electrónica de la proteína *1lza* (izquierda). Los puntos con mayor densidad electrónica se muestran en rojo, con menor densidad en azul. Nótese que, al realizarse la representación usando como origen el centro del objeto tridimensional, las capas centrales tienen una mayor densidad de puntos que las capas más exteriores. Cada capa esférica de este muestreo radial se emplea posteriormente para realizar la expansión sobre la base de esféricos armónicos.

De este modo, el método de ajuste FRM3 realiza la exploración completa del espacio de búsqueda siguiendo el siguiente esquema (ver **Fig. 4.4**):

- *Pre-cálculo de coeficientes* d^l_{mn}.
- *Pre-cálculo de expansión sobre esféricos armónicos de g con origen en su propio centro.*
- *Muestreo sistemático traslacional. Para cada posición traslacional:*
  - *Cálculo de la expansión sobre esféricos armónicos de f con origen determinado por la posición del centro de g.*
  - Cálculo de $I^l_{mm'}$ (ver ecuación (4.10)).
  - *Cálculo de la correlación rotacional (ver ecuación (4.16)).*
  - *Introducción ordenada de las mejores conformaciones rotacionales encontradas en una lista de soluciones.*
- *Devolución de la lista de soluciones ordenada en función de la correlación*.

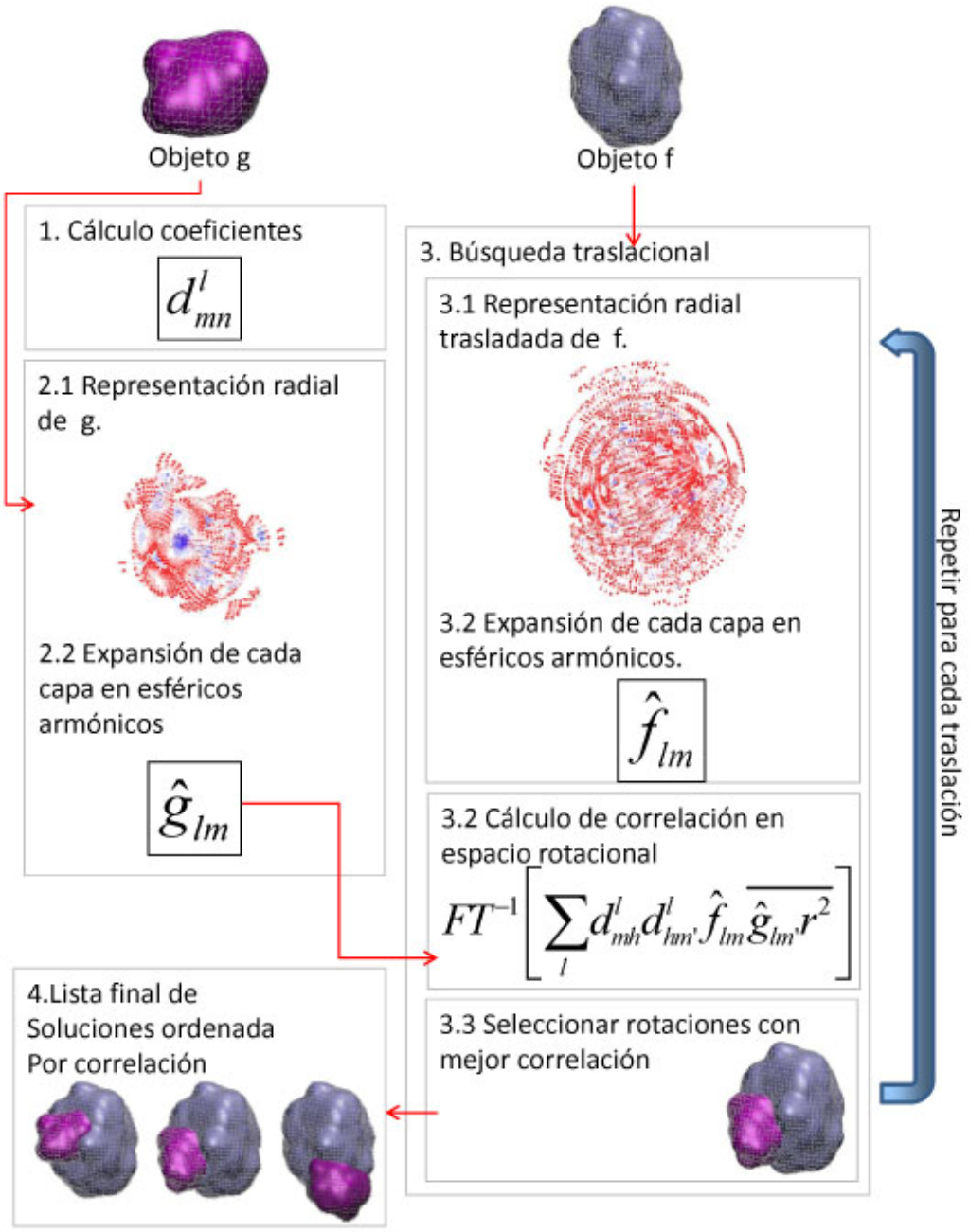

Fig.4.4- Esquema de realización de un proceso de Ajuste Rotacional Rápido (ADP_EM). La expansión armónica de uno de los objetos sólo ha de realizarse una vez ya que éste permanece fijo. La expansión armónica del segundo objeto se genera para cada traslación variando el origen de la representación esférica. La secuencia en la que las acciones son realizadas es indicada por la numeración establecida. Las líneas rojas indican el flujo de datos entre las distintas acciones.

4.1.2 Ajuste rotacional rápido sobre cinco grados de libertad (FRM5)

En lugar de emplear tres dimensiones cartesianas (traslación) y tres dimensiones angulares (rotación) para describir el espacio 6-Dimensional, es posible describirlo mediante dos rotaciones, una para cada objeto a ajustar, y una distancia cartesiana que define la distancia relativa entre ellos. Las dos rotaciones

y la distancia definen seis grados de libertad, uno cartesiano (distancia) y cinco angulares, ya que uno de los ángulos es redundante, siendo empleado para definir ambas rotaciones (ver **Fig. 4.5**). El hecho de describir cinco de los grados de libertad mediante variables angulares posibilita extender la aceleración realizada anteriormente en FRM3 a dos grados de libertad más [130].

Siguiendo este razonamiento, se define la correlación de dos matrices tridimensionales, f y g, en términos de dos rotaciones $R=\{\varphi,\phi,\psi\}$ y $R'=\{\varphi',\phi',\psi'\}$ y una distancia relativa entre los objetos ρ:

$$c(R,R',\rho)=\sum_{r,\beta,\lambda}\overline{\Lambda_R f(r,\beta,\lambda)}\cdot\overline{T_\rho\Lambda_{R'}g(r,\beta,\lambda)} \qquad (4.18)$$

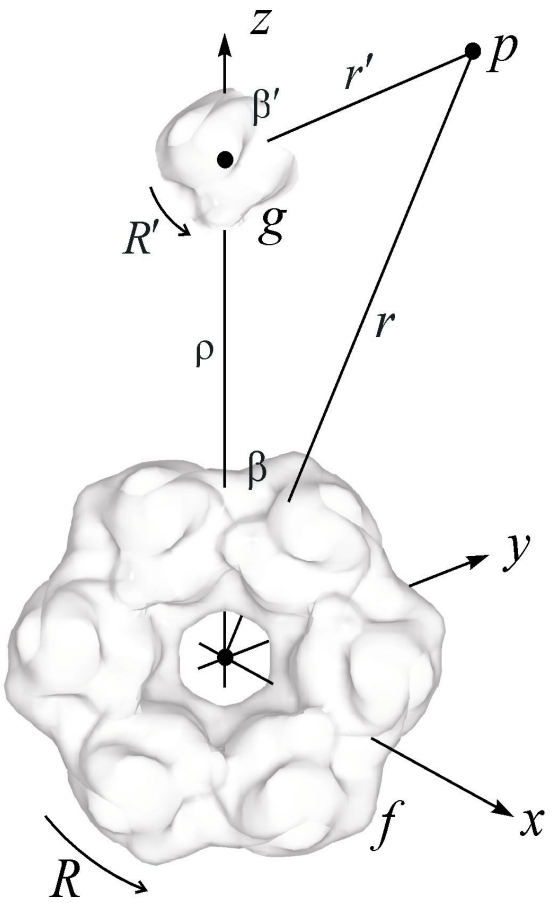

Fig.4.5- Exploración del espacio 6-Dimensional mediante dos rotaciones y una traslación. R y R' son las rotaciones de los objetos *f* y *g* respecto a su centro de coordenadas. ρ define la translación entre ambos objetos sobre el eje Z. Nótese que las rotaciones R y R' definen cinco grados de libertad ya que uno de los ángulos de rotación relativos es redundante. En este caso, el espacio de búsqueda queda definido por cinco grados de libertad rotacionales y uno traslacional.

A ambos factores se les aplica el complejo conjugado con el fin de obtener el signo correcto en la exponencial de la transformada de Fourier. T_ρ es un operador de traslación definido como:

$$(T_\rho g)(r,\beta,\lambda)=g(\sqrt{r^2-2r\rho\cos\beta},\frac{(r\cos\beta-\rho)}{\sqrt{r^2-2r\rho\cos\beta}},\lambda)=g(\tilde{r},\tilde{\beta},\tilde{\lambda}) \qquad (4.19)$$

De esta forma y teniendo en cuenta la ortogonalidad de los esféricos armónicos se obtiene:

$$c(R,R',\rho)=\sum_{ll'mm'n}\left[\overline{D^l_{nm}(R)D^{l'}_{nm'}(R')}\sum_{\beta,\lambda}\overline{\hat{f}_{lm}(r)\hat{g}_{l'm'}(\tilde{r}')Y_{\ln}(\beta,\lambda)Y_{l'n'}(\tilde{\beta}',\tilde{\lambda}')}\right]=$$
$$\sum_{ll'mm'n}\left[\overline{D^l_{nm}(R)D^{l'}_{nm'}(R')}\sqrt{(l+1/2)(l'+1/2)}\sum_\beta d^l_{n0}(\beta)\sin\beta\left[\sum_r\overline{\hat{f}_{lm}(r)\hat{g}_{l'm'}(\tilde{r}')}d^{l'}_{n0}(\tilde{\beta}')r^2\right]\right] \qquad (4.20)$$

Definiendo la integral entre los coeficientes armónicos como:

$$I^{ll'}_{mnm'}(\rho)=\sqrt{(l+1/2)(l'+1/2)}\sum_{\beta} d^{l}_{n0}(\beta)\sin\beta\left[\sum_{r}\overline{\hat{f}_{lm}(r)\hat{g}_{l'm'}(\tilde{r}')}d^{l'}_{n0}(\tilde{\beta}')r^{2}\right] \tag{4.21}$$

La correlación en función de las rotaciones y la distancia se simplifica a:

$$c(R,R',\rho)=\sum_{ll'mm'n}\left[\overline{D^{l}_{nm}(R)D^{l'}_{nm'}(R')}I^{ll'}_{mnm'}(\rho)\right] \tag{4.22}$$

Las rotaciones son factorizadas de forma equivalente a como se hizo para FRM3 (4.12) de forma que:

$$\begin{aligned}&\xi=\varphi-\pi/2,\quad \eta=\pi-\phi,\quad \omega=\psi-\pi/2\\&\xi'=\varphi'-\pi/2,\quad \eta'=\pi-\phi',\quad \omega'=\psi'-\pi/2\end{aligned} \tag{4.23}$$

Aplicando (4.14) y definiendo $\sigma=\varepsilon-\varepsilon'$se obtiene:

$$\begin{aligned}&c(R,R',\rho):=T(\sigma,\eta,\omega,\eta'\omega';\rho)=\\&\sum_{ll'mm'hh'}\left[(-1)^{n}d^{l}_{nh}(\pi/2)d^{l}_{hm}(\pi/2)d^{l'}_{-nh'}(\pi/2)d^{l'}_{h'm'}(\pi/2)e^{i(n\sigma+h\eta+m\omega+h'\eta'+m'\omega')}I^{ll'}_{mnm'}(\rho)\right]\end{aligned} \tag{4.24}$$

Nótese que mediante la introducción del ángulo σ ambas rotaciones pueden definirse con un ángulo común, reduciéndose el número de grados de liberad rotacionales de seis a cinco. Por otro lado, y tal y como ocurría en FRM3, el cálculo de la correlación sobre todo el espacio de búsqueda pasa a requerir únicamente coeficientes d^{l}_{hm}definidos para $\pi/2$.

Por último, la correlación puede ser calculada a partir de su transformada de Fourier:

$$\begin{aligned}&c(R,R',\rho):=FFT^{-1}(\hat{T}(m,n,h,m',h';\rho))=\\&FFT^{-1}((-1)^{n}\sum_{ll'}d^{l}_{nh}(\pi/2)d^{l}_{hm}(\pi/2)d^{l'}_{-nh'}(\pi/2)d^{l'}_{h'm'}(\pi/2)I^{ll'}_{mnm'}(\rho))\end{aligned} \tag{4.25}$$

FRM5 aporta las mismas ventajas que FRM3 extrapoladas a dos grados de libertad más, resultando de este modo un método más rápido. Sin embargo, presenta una importante desventaja: el cálculo de la integral $I^{ll'}_{mnm'}(\rho)$ es mucho más complejo, requiriendo almacenar en memoria una matriz 5-Dimensional de valores complejos. En el caso de FRM3 este número de valores viene determinado por los coeficientes *m*, *h* y *m'* (véase (4.16)) que están definidos en función del número de armónicos esféricos empleados en la representación (*B*). Por el contrario, en FRM5 el número de elementos que se deben mantener en memoria depende de los coeficientes n, *h*, *m*, *h'* y *m'* (véase (4.25)). Debido a las dependencias de estos coeficientes en *B,* el número de elementos en cada matriz es:

$$N_{FRM3} = \sum_{m=-B}^{B} \sum_{h=-B}^{B} \sum_{m=0}^{B} 1 = (2B+1)(2B+1)(B+1) = (2B+1)^2(B+1) \quad (4.26)$$

$$N_{FRM5} = \sum_{n=-B}^{B} \sum_{h=-B}^{B} \sum_{m=0}^{B} \sum_{h'=0}^{B} \sum_{m'=0}^{B} 1 = (2B+1)^5(B+1) \quad (4.27)$$

Por tanto, los requerimiento de memoria de FRM3 escalan en función de O(B^3) frente a O(B^5) del método FRM5. Por ejemplo, para *B=32* (valor que supone un muestreo rotacional de menos de 6°) con FRM3 es necesario almacenar una matriz de 139.425 elementos complejos, mientras que con FRM5 son necesarios 38.289.590.625 elementos. Esta gran cantidad de información hace que los requisitos de memoria RAM para esta aplicación sean excesivos, no pudiendo garantizarse en computadores de tamaño normal. Así, se han realizado desarrollos experimentales para comprobar las limitaciones de FRM5, obteniéndose que la base *B* de esféricos armónicos empleada debe mantenerse en valores pequeños con el fin de no saturar la memoria de los sistemas. Esta reducción de *B* implica una limitación importante en el muestreo del espacio de búsqueda que no garantiza en la mayoría de los casos una correcta exploración. Por ello, el uso de este método ha sido descartado en favor de FRM3 que, si bien no presenta una aceleración tan grande, muestra una relación entre velocidad, fiabilidad y requerimientos de memoria muy aceptable.

4.2 Desarrollo de aplicaciones bioinformáticas

En esta sección se describe el uso de FRM3 para la resolución de los dos tipos de problemas de ajuste previamente expuestos. En una primera sección se detalla la aplicación creada para resolver casos de ajuste multi-resolución mientras que en la segunda sección se realiza lo propio para casos de ajuste proteína-proteína.

4.2.1 Adaptación de FRM3 al Ajuste Multi-resolución: ADP_EM

El ajuste 6-Dimensional rígido a diferentes resoluciones se emplea para interpretar mapas de densidad electrónica a baja resolución de estructuras macromoleculares, obtenidos mediante métodos EM, a partir de estructuras atómicas de sus componentes [84, 131]. Como ya se ha indicado, el conocimiento de la estructura atómica de complejos macromoleculares puede ser de gran utilidad. Sin embargo, la obtención de estas grandes estructuras mediante técnicas que proporcionan una alta resolución (cristalografía de rayos X, resonancia magnética) es compleja y no siempre posible. Técnicas como EM

presentan menos limitaciones para visualizar complejos macromoleculares grandes pero, como contrapartida, producen estructuras con una resolución muy baja, no permitiendo determinar localizaciones concretas de átomos. EM es además una técnica muy versátil, permitiendo visualizar macromoléculas en distintos estados o conformaciones funcionales en solución. En este contexto, la información proporcionada por EM puede ser combinada con el conocimiento de las estructuras atómicas de las proteínas que componen un complejo con el fin de modelar su estructura atómica. Esta combinación es realizada mediante el ajuste interno de las estructuras atómicas de los componentes dentro de la estructura o mapa a baja resolución, como si de un puzle se tratara (ver **Fig. 4.6**). Además, el empleo de herramientas bioinformáticas de modelado posibilita este ajuste incluso cuando no se dispone de estructuras atómicas de los componentes. Herramientas bioinformáticas de modelado por homología [132] permiten modelar la estructura atómica a partir de estructuras de moléculas homólogas, esto es, con una gran similitud en su secuencias de aminoácidos y que en principio deben presentar un plegamiento similar. Debido al amplio número de estructuras homólogas que estos métodos generan así como a las distintas aproximaciones alternativas para realizar este modelado, el número de posibles ajustes es muy elevado, aumentando la complejidad computacional.

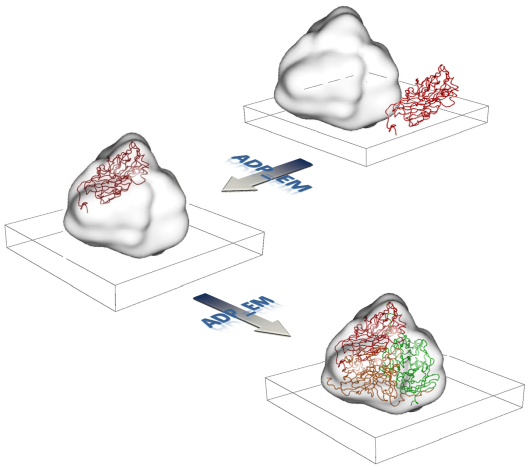

Fig.4.6- Modelado de la estructura atómica de un complejo macromolecular mediante ajuste multi-resolución, realizada a partir de la información a baja resolución del complejo (volumen transparente) y de las estructuras atómicas de sus componentes (representaciones en cinta). La aplicación ADP_EM ajusta la estructura atómica del componente a las zonas del mapa a baja resolución donde la correlación de densidades electrónicas es mayor. En el ejemplo mostrado, al ser el complejo un trímero (conformado por tres estructuras idénticas) existen tres máximos de correlación.

Por otra parte, desde el punto de vista de modelado, este tipo de ajuste multi-resolución puede ser también empleado para evaluar los modelos de homología. En otras palabras, permite determinar qué métodos de homología proporcionan modelos más cercanos a la realidad experimental (ver **Fig. 4.7**).

A continuación se describen los aspectos del método FRM3 que han sido adaptados para resolver el problema de ajuste multi-resolución. Esta adaptación

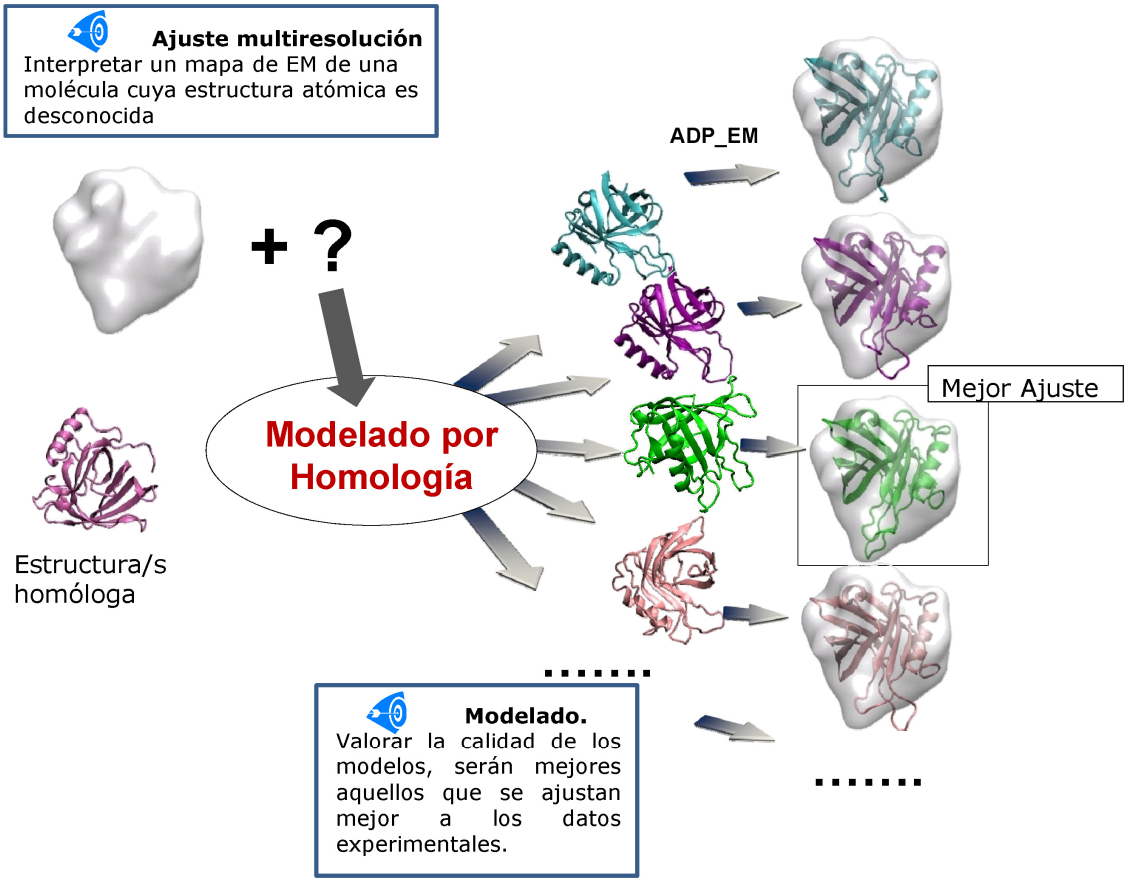

Fig.4.7- Combinación de ajuste multi-resolución con herramientas de modelado por homología. Desde el punto de vista de ajuste, el modelado por homología proporciona modelos que pueden ser empleados en el ajuste multi-resolucion. Desde el punto de vista de modelado, el ajuste multi-resolución permite evaluar los modelos creados por herramientas de modelado, determinando el modelo que más se aproxima a la estructura real.

ha dado lugar a la creación de la aplicación denominada ADP_EM. Básicamente, tal y como se indicó al describir FRM3, la caracterización del método vendrá dada por el tipo de función de correlación empleada así como por las propiedades de los objetos correlacionadas. Así mismo, la elección del origen de las expansiones armónicas de los objetos juega un importante papel a la hora de realizar el cálculo de la función de correlación. Otro aspecto muy importante de cara a obtener una búsqueda eficiente concierne a la forma en que el espacio traslacional es restringido *a priori* para evitar explorar conformaciones sin sentido físico. Por último, se presentará un esquema de la ejecución de la aplicación ADP_EM en el que se muestra la combinación de todas las adaptaciones descritas.

4.2.1.1 Función de Correlación

Tal y como se ha indicado, el ajuste multi-resolución es de tipo interno. Dado que el objetivo es la determinación de la posición de una estructura atómica dentro de un mapa tridimensional de densidad electrónica, la correlación se

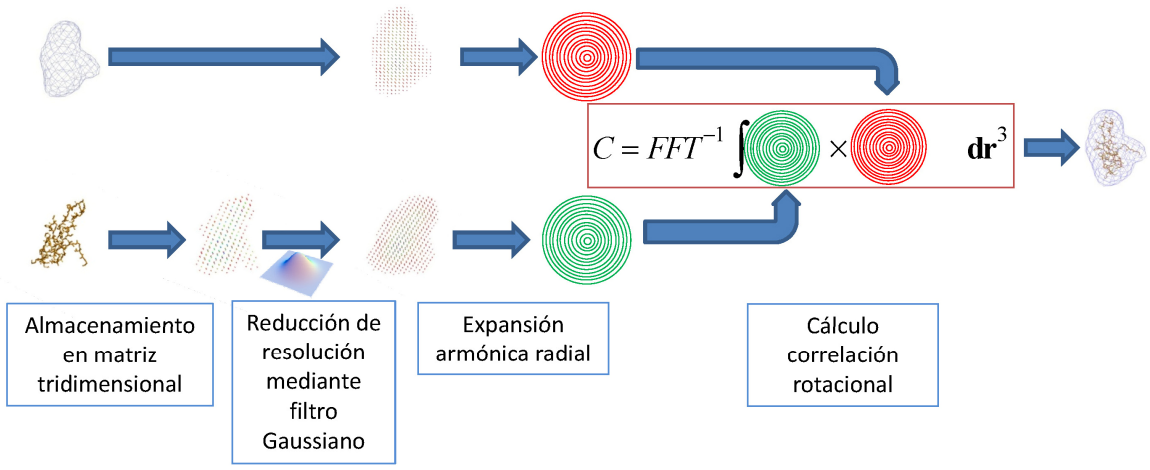

Fig.4.8- Ilustración del cálculo de correlación empleado en el ajuste multi-resolución. La densidad electrónica del mapa experimental almacenada en una matriz tridimensional se expande en funciones armónicas de forma radial. En el caso de la estructura atómica, la masa electrónica de los átomos se proyecta sobre una matriz tridimensional que posteriormente es filtrada para reducir su resolución de acuerdo a la del mapa experimental. Tras el filtrado se realiza la expansión armónica. Ambas expansiones armónicas son empleadas para calcular la correlación.

calculará como el producto escalar de las densidades electrónicas de ambos elementos. El criterio que determina el mejor ajuste será una maximización de dicha correlación. Por tanto, en la ecuación (4.1) para el cálculo de la correlación en el método general, las matrices f y g se corresponden a proyecciones de la densidad electrónica de los elementos a ajustar. En el caso del mapa de baja resolución, la propia técnica EM devuelve una matriz con los valores de densidad del complejo. En el caso de la estructura atómica, las densidades electrónicas de sus átomos son proyectadas en una matriz tridimensional que posteriormente es filtrada para obtener una resolución equivalente a la del mapa. A partir de estas matrices se realizan expansiones radiales sobre una base de esféricos armónicos que son empleadas en el cálculo de la correlación (ver **Fig. 4.8**).

Una variante al simple cálculo del producto escalar para determinar el mejor ajuste consiste en realizar un filtrado Laplaciano sobre los mapas de densidad. Mediante este filtrado se obtiene mayor contraste sobre los contornos de los mapas, mejorándose el ajuste de las superficies de los objetos. Por tanto, si bien este filtrado puede proporcionar mejores ajustes para estructuras atómicas localizadas en la superficie de la macromolécula, los ajustes no serán tan buenos cuando dichas estructuras se encuentren en el interior de ésta. Por tanto, la aplicación del filtrado Laplaciano es opcional en la aplicación ADP_EM, permitiendo al usuario emplearlo en función de la naturaleza del problema.

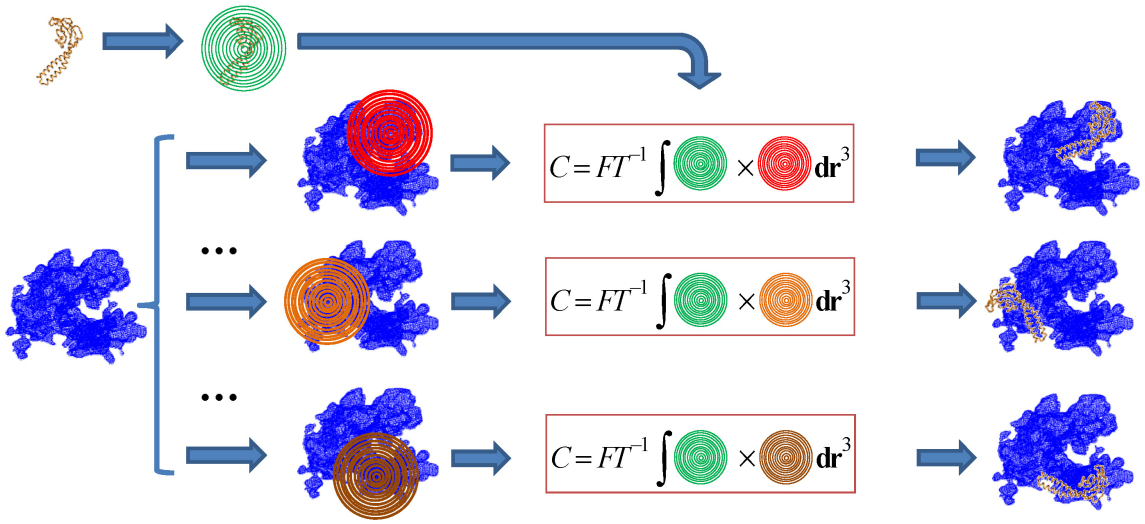

Fig.4.9- Esquema ilustrativo de la determinación del origen para las expansiones armónicas radiales en cada punto traslacional en el ajuste multi-resolución. Al permanecer fija la estructura (objeto más pequeño) sólo es necesario realizar una vez su expansión armónica estableciendo su origen en el centro de masas. Para el mapa a baja resolución (objeto más grande) se realiza una expansión armónica radial en cada traslación explorada. En cada una de estas traslaciones, el origen de la expansión se establecerá en la posición donde se fija la estructura. Las expansiones radiales sobre el mapa están limitadas a un número de capas igual al necesario para representar a la estructura. Estas expansiones armónicas son empleadas para determinar la mejor rotación de la estructura en cada punto del mapa.

4.2.1.2 Origen de las expansiones armónicas

Al realizar el cálculo de correlación, los orígenes de las expansiones radiales armónicas se consideran superpuestos. De esta forma, el desplazamiento entre ambos elementos se consigue variando el origen en uno de ellos. En principio, determinar uno u otro objeto como fijo (su expansión radial se mantiene invariable sobre su centro de masas) y móvil (el origen de la expansión se altera) puede resultar indiferente, sin embargo si se tiene en cuenta la naturaleza de una representación radial armónica, resulta más efectivo fijar el objeto más pequeño (ver **Fig. 4.9**). Como ya se indicó al final de la sección **4.1.1**, la representación obtenida por la expansión armónica emplea el mismo número de esféricos armónicos para representar todas las capas que la componen, lo que conlleva una pérdida de representatividad para las capas más alejadas. Dicho de otra forma, el objeto tridimensional está mejor aproximado en las capas más cercanas al origen. Por ello, fijando el origen de las representaciones en el objeto más pequeño reducirá este tipo de distorsiones. En el caso del ajuste multi-resolución la estructura atómica siempre tendrá dimensiones menores o como mucho iguales al mapa, por lo tanto se ha optado por mantener dicha estructura como objeto fijo, creándose su representación esférica una única vez, mientras que la representación del mapa se establece múltiples veces modificando su origen para

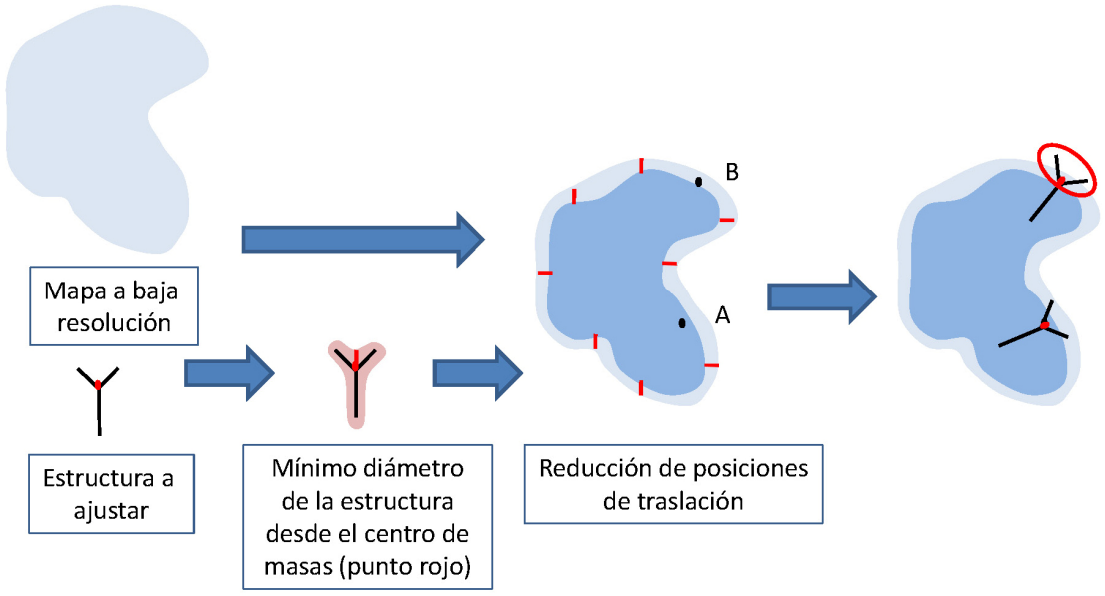

Fig.4.10- Delimitación del espacio traslacional en el ajuste externo. Las posiciones dentro del mapa a baja resolución son reducidas en función del diamétro mínimo de la estructura atómica. La región resultante (en azul oscuro) define posiciones válidas de búsqueda, como (A), en las que la estructura a ajustar está contenida dentro del mapa de baja resolución al menos para una rotación. En posiciones no validas como (B) la estructura sobresale parcialmente del mapa para toda posible rotación, no proporcionándose ajustes válidos.

realizar traslaciones. Nótese que para cada traslación la representación armónica del mapa está limitada por el diámetro máximo de la estructura, no teniendo sentido crear más capas puesto que no existen capas de la estructura con las que correlacionarlas. Por tanto, mantener fijo la estructura atómica proporciona además una minimización del cálculo de correlación.

4.2.1.3 Delimitación del espacio traslacional

Al ser un problema de ajuste interno, es decir uno de los objetos debe de posicionarse dentro del otro, el espacio traslacional se puede limitar fácilmente. Las posiciones con valores no nulos del objeto más grande, en este caso el mapa a baja resolución, determinan las posiciones sobre las que *a priori* la estructura atómica puede ser colocada. Sin embargo en algunas de estas posiciones la estructura mostrará, sea cual sea su rotación, partes fuera del mapa. Para evitar este tipo de situaciones sin sentido físico, la máscara de posiciones inicial es *erosionada* en función del diámetro mínimo de la estructura. El diámetro mínimo se define como la distancia desde el centro de la estructura al átomo de su superficie más cercano al centro. De esta forma se garantiza que posiciones traslacionales que no presentan al menos una rotación en la que la estructura esté completamente introducida en el mapa serán descartadas (ver **Fig. 4.10**).

Sin embargo, es necesario tener en cuenta que esta delimitación presentará problemas en su aplicación sobre casos en los que las estructuras a ajustar presenten zonas centrales vacías.

Por otro lado, el caso en el que la estructura a ajustar no sea un elemento sino que se corresponda con toda la molécula (caso que se produce con modelos por homología) permite realizar una exploración eficaz de las traslaciones. Al ajustarse la estructura completa, su localización debe lógicamente encontrarse en la zona central del mapa. Si se comienza la búsqueda en las posiciones centrales de la máscara y se continúa ampliándola de forma radial, es esperable que los valores de ajuste encontrados vayan decreciendo. Por tanto, si se establece un límite proporcional al mejor valor de ajuste encontrado, la búsqueda puede detenerse cuando los valores encontrados en un determinado radio sean menores que ese límite, no teniéndose así que explorar todos los puntos de traslación.

Teniendo en cuenta estas consideraciones, la aplicación ADP_EM ha sido realizada con distintas opciones de exploración. Junto a una exploración sin restricciones del espacio traslacional entre los dos elementos, se han añadido una búsqueda limitada por la reducción de la máscara de posiciones y una búsqueda radial iniciada en el centro del mapa.

4.2.1.4 Esquema general de funcionamiento de ADP_EM

Aplicando todas las adaptaciones descritas en los apartados anteriores sobre el método FRM3, el esquema de funcionamiento de ADP_EM es el ilustrado en **Fig. 4.11**. Para una descripción más detallada de la aplicación véase [133].

4.2.2 Adaptación de FRM3 al Ajuste Proteína-Proteína: FRODOCK

Como se ha descrito en la sección **2.2.1.4**, la determinación de cómo dos proteínas interactúan para conformar un complejo se realiza mediante la búsqueda del ajuste externo que minimiza la correlación entre términos energéticos de ambas proteínas.

La combinación de la expresión de términos energéticos en matrices tridimensionales junto con la eficiencia del método FRM3 es especialmente ventajosa para resolver este tipo de problemas. En las siguientes secciones se muestran las adaptaciones realizadas en el método de ajuste así como un esquema general de la aplicación resultante, denominada FRODOCK (*Fast ROtational DOCKing*). Es importante resaltar que estas adaptaciones han

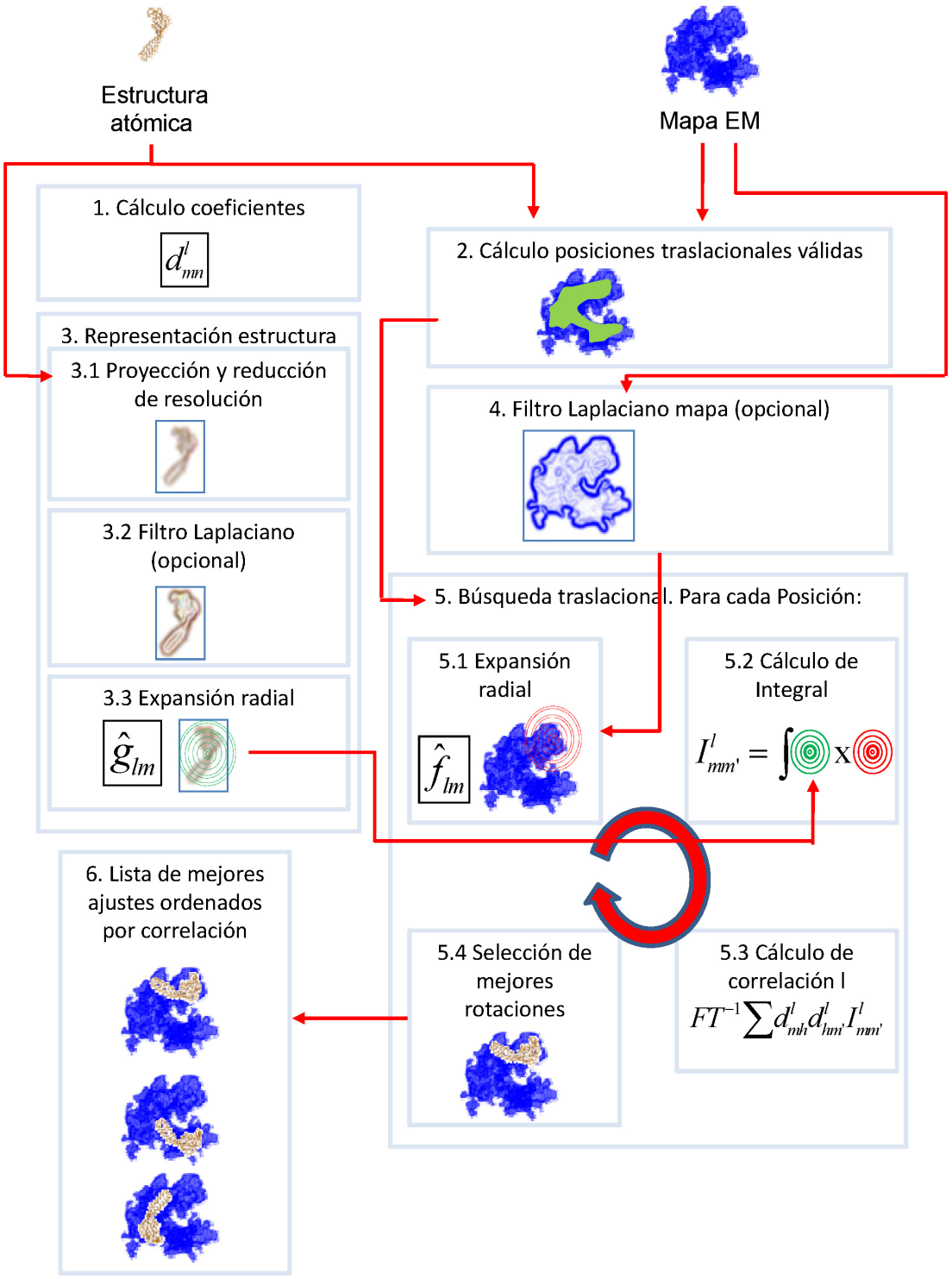

Fig.4.11- Esquema de funcionamiento de la aplicación ADP_EM. La secuencia en la que las acciones son realizadas es indicada por la numeración establecida. Las líneas rojas indican el flujo de datos entre las distintas acciones.

requerido la reformulación y desarrollo de nuevas expresiones matemáticas para poder aplicar los conceptos de FRM al ajuste proteína-proteína.

4.2.2.1 Función de Correlación

La formación de un complejo entre dos proteínas depende de la estabilidad energética de la conformación establecida entre ambas. Dicha estabilidad es inversamente proporcional a la energía libre del sistema. Por tanto, el criterio de ajuste empleado deberá de minimizar la combinación de las energías presentes en la interacción. Entre los posibles tipos de interacciones energéticas que pueden contribuir a la estabilidad del complejo se encuentran la interacción producida por la energía de Van der Waals [134-135], la interacción electromagnética [135-138], la interacción asociada a procesos de desolvatación [110], establecimiento de puentes de hidrógeno, etc. En el caso de FRODOCK han sido empleados los términos energéticos asociados a Van der Waals, electroestática y desolvatación por ser los más importantes en la interacción proteína-proteína y ser fácilmente representables en matrices tridimensionales.

Para cada uno de estos términos, en una traslación dada, la energía presente en la interacción se calcula mediante una función de correlación en la que los términos *f* y *g* de la integral (4.1) son respectivamente un potencial energético y una propiedad escalar. Así, una matriz tridimensional representa el potencial energético de la proteína receptor (usualmente la proteína de mayor tamaño) mientras que otra representa una propiedad escalar (masa, carga, etc.) de cada átomo de la proteína ligando (proteína de menor tamaño). El potencial energético de la proteína receptor en un punto se define como la combinación de fuerzas producidas por todos los átomos de la proteína sobre un elemento diferencial situada en dicho punto. Por su parte, la proteína ligando se representa mediante la proyección de la propiedad escalar L (masa, carga, etc.):

$$g(r,\beta,\lambda)=\sum_{i=0}^{N} L_i \cdot \delta_{r,\beta,\lambda}(r_i,\beta_i,\lambda_i) \tag{4.28}$$

Siendo N el número de átomos del ligando, L_i la propiedad escalar del átomo *i* y δ la función de Kronecker definida como:

$$\delta_{r,\beta,\lambda}(r_i,\beta_i,\lambda_i)=\begin{cases}1 & ,r_i=r\wedge\beta_i=\beta\wedge\lambda_i=\lambda\\ 0 & ,r_i\neq r\vee\beta_i\neq\beta\vee\lambda_i\neq\lambda\end{cases} \tag{4.29}$$

Sustituyendo (4.28) en la ecuación (4.2), pueden calcularse los coeficientes armónicos esféricos asociados $\hat{g}_{lm}$:

$$\hat{g}_{lm}(r) = \sum_{\beta\gamma}\left[\sum_{i=0}^{N} L_i \cdot \delta_{r,\beta,\lambda}(r_i,\beta_i,\lambda_i)\overline{Y_{l,m}(\beta,\lambda)}\right] = \sum_{i=0}^{N} L_i \sum_{\beta\gamma} \delta_{r,\beta,\lambda}(r_i,\beta_i,\lambda_i)\overline{Y_{l,m}(\beta,\lambda)} = \sum_{i=0}^{N} \frac{L_i}{r^2}\delta_r(r_i)\overline{Y_{l,m}(\beta_i,\lambda_i)} \quad (4.30)$$

Sustituyendo en la función de correlación rotacional en el espacio de Fourier (4.16):

$$\hat{C}(m,h,m') = \sum_{lmm'}\left[\sum_{h}\left[d^l_{mh}(\pi/2)d^l_{hm'}(\pi/2)\right]\sum_{r}\left[\hat{f}_{lm}(r)\sum_{i=0}^{N} -L_i\delta_r(r_i)Y_{l,m'}(\beta_i,\lambda_i)\right]\right] = \sum_{lmm'}\left[\sum_{h}\left[d^l_{mh}(\pi/2)d^l_{hm'}(\pi/2)\right]\sum_{i=0}^{N}\left[-L_iY_{l,m}(\beta_i,\lambda_i)\hat{f}_{lm'}(r_i)\right]\right] \quad (4.31)$$

Obsérvese que en la función de correlación la sustitución ha sido realizada invirtiendo el término. Debido a que la reducción de la energía libre repercute en la estabilidad del sistema, el criterio de ajuste se orienta a buscar valores negativos de correlación. Cambiando el signo de uno de los términos se invierte el valor de correlación resultante, de forma que el criterio de ajuste pasa a ser una maximización de la correlación. En esta ecuación se evita el cálculo implícito de los coeficientes esféricos asociados a g, necesitándose únicamente conocer los valores de los armónicos esféricos. Además, teniendo en cuenta que los valores $\hat{f}_{lm}(r_i)$ del receptor sólo se calculan para un conjunto finito de capas, resulta más eficiente aproximar el valor de la correlación mediante la agrupación de átomos en capas. Los átomos que presenten un valor de r dentro de un determinado intervalo son adscritos a una capa. Así:

$$\sum_{i=0}^{N}\left[-L_iY_{l,m'}(\beta_i,\lambda_i)\hat{f}_{lm}(r_i)\right] \approx \sum_{j=0}^{C}\hat{f}_{lm}(\overline{r_j})\sum_{k=0}^{n_{C_j}}\left[-L_{C_jk}Y_{l,m'}(\beta_{C_jk},\lambda_{C_jk})\right] \quad (4.32)$$

Siendo *C* el conjunto finito de capas, $\overline{r_j}$ el valor medio de los radios de todos los átomos en la capa *j* y C_j el subconjunto de átomos en la capa *j*. Sustituyendo en la función de correlación rotacional (4.16) se obtiene:

$$F^{-1}\left[\hat{C}(m,h,m')\right] = F^{-1}\left[\sum_{l} d^l_{mh}(\pi/2)d^l_{hm'}(\pi/2)I^l_{mm'}\right] \quad (4.33)$$

Siendo la integral*:*

$$I^l_{mm'} = \sum_{j=0}^{C}\left[\hat{f}_{lm}(\overline{r_j})\sum_{k=0}^{n_{C_j}}\left[-L_{C_jk}Y_{l,m}(\beta_{C_jk},\lambda_{C_jk})\right]\right] \quad (4.34)$$

Mediante este nuevo desarrollo se obtiene la correlación para un determinado término energético de forma eficiente. Así, el criterio de ajuste viene definido por la maximización de la sumatoria, adecuadamente ponderada por los valores w, de las correlaciones obtenidas para las diferentes interacciones energéticas:

$$C_{Total} = w_1 C_1 + w_2 C_2 + ... + w_n C_n = w_1 F^{-1}[\hat{C}_1] + w_2 F^{-1}[\hat{C}_2] + ... + w_n F^{-1}[\hat{C}_n] \quad (4.35)$$

Por último, aplicando las propiedades de la transformada (4.17) pueden obtenerse el valor de criterio de ajuste mediante una sola transformada inversa de Fourier:

$$C_{Total} = FFT^{-1}\left[w_1 \hat{C}_1 + w_2 \hat{C}_2 + ... + w_n \hat{C}_n \right] \quad (4.36)$$

Y como las diferentes ecuaciones $\hat{C}$ (4.33) sólo varían en el valor de la integral $I^l_{mm'}$, se obtiene:

$$C_{Total} = FFT^{-1}\left[\sum_l d^l_{mh}(\pi/2) d^l_{hm'}(\pi/2) \left[w_1 \mathbf{I}_{\mathbf{1}}{}^l_{mm'} + w_2 \mathbf{I}_{\mathbf{2}}{}^l_{mm'} + ... + w_n \mathbf{I}_{\mathbf{n}}{}^l_{mm'} \right] \right] \quad (4.37)$$

A continuación se detallan las características de los tres términos energéticos empleados en el cálculo de la correlación total.

Energía de Van der Waals

Las interacciones de van der Waals definen las fuerzas de repulsión y atracción entre átomos. Este tipo de interacción proporciona información sobre el grado de complementariedad de las superficies de contacto entre las proteínas, siendo por ello una interacción muy importante en el cálculo del ajuste. Las interacciones de van der Waals son descritas mediante el potencial Leonard-Jones 6-12. Sin embargo, la parte repulsiva de este potencial (producida a distancias cortas entre átomos) es truncada para reducir la sensibilidad a los cambios conformacionales, permitiéndose una ligera tolerancia a la flexibilidad de las proteínas. Este potencial suavizado ha sido previamente usado en ajuste proteína-proteína [139]. De esta forma la matriz tridimensional que representa el potencial energético del receptor se construye calculando, para cada posición del mapa, la combinación de fuerzas producidas por todos los átomos del receptor sobre un átomo genérico en el caso de que fuera colocado en esa posición

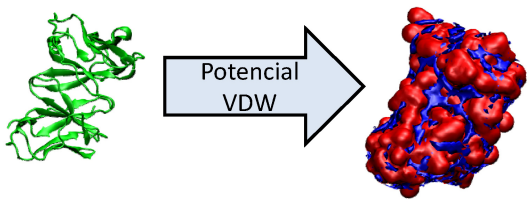

Fig.4.12- Potencial de van der Waals (superficies sólidas) creado a partir de la estructura atómica *HyHel-5* (diagrama de cintas cerde). Zonas con potencial positivo se muestran en rojo, las negativas en azul.

(**Fig. 4.12**). Así, para cada posición se calcula:

$$P_w(p) = \sum_i^N P_W^i(p) \tag{4.38}$$

Siendo $P_W^i(p)$:

$$P_W^i(p) = \begin{cases} \hat{P}_W^i(p) & ,\hat{P}_W^i(p) \le 0 \\ \dfrac{\hat{P}_W^i(p) P_{\max}}{\hat{P}_W^i(p) + P_{\max}} & ,\hat{P}_W^i(p) > 0 \end{cases} \quad ,\hat{P}_W^i(p) = \frac{A_i}{r_{p_i}^6} + \frac{1}{r_{p_i}^{12}} \tag{4.39}$$

Donde N es el conjunto de átomos pesados del receptor, r_{p_i} es la distancia del átomo *i* a la posición *p*, $P_{\max}$ es el límite establecido para las fuerzas repulsivas (positivas) y A_i es un valor constante para cada tipo de átomo. Así, los coeficientes $\hat{f}_{lm}$ en la ecuación de correlación (4.33) se calculan a partir de este potencial mientras que el término asociado al ligando es calculado empleando la masa electrónica como propiedad escalar L .

Energía de Electroestática

La interacción electroestática está presente en todos los procesos a nivel molecular. De modo similar al potencial de van der Waals, el mapa del potencial electroestático del receptor (**Fig. 4.13**) es calculado en cada posición agregando las fuerzas electroestáticas producidas por todos sus átomos sobre un átomo genérico de carga 1 colocado en dicha posición:

$$P_E(p) = \sum_i^N \hat{P}_E(p) \quad ,\hat{P}_E(p) = \frac{q_i}{4r_{pi}^2} \tag{4.40}$$

Siendo q_i la carga electroestática asociada al átomo *i*. Por último, los valores del potencial son limitados dentro de un rango de ±10 *Kcal/mol e* para evitar la aparición de fuerza electroestáticas poco realistas debido a la suavización del potencial de van der Waals. A partir de este potencial son calculados los coeficientes $\hat{f}_{lm}$ que determinan la expansión armónica del receptor. Este potencial del receptor es correlacionado con una proyección de las cargas electroestáticas asociadas con los átomos del ligando, siendo la

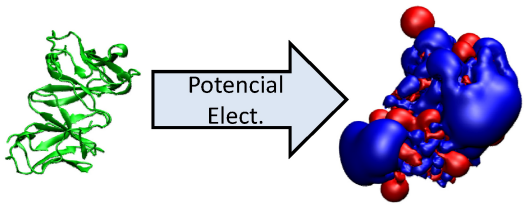

Fig.4.13- Potencial Electroestático (superficies sólidas) creado a partir de la estructura atómica *HyHel-5* (diagrama de cintas cerde). Zonas con potencial positivo se muestran en rojo, las negativas en azul.

carga electroestática la propiedad escalar L empleada.

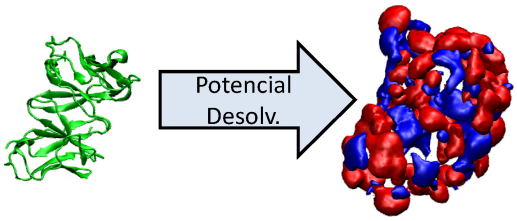

Fig.4.14- Potencial de desolvatación (superficies sólidas) creado a partir de la estructura atómica *HyHel-5* (diagrama de cintas cerde). Zonas con potencial positivo se muestran en rojo, las negativas en azul.

Energía de desolvatación

La energía de desolvatación se define como la ganancia obtenida al desplazar moléculas de agua en la interfaz establecida entre las proteínas (**Fig. 4.14**). Aquí ha sido calculado como la suma de las contribuciones proporcionales de cada átomo al área de la superficie de solvatación cubierta en la interacción (BSA del inglés *Buried Solvation Area*), de forma que el potencial de desolvatación para el receptor en una posición se calcula:

$$P_s(p) = \sum_i^N BSA_i(p)\sigma_i \qquad (4.41)$$

Siendo σ_i el parámetro de solvatación para el átomo *i* tabulado según [140-141]. Para estimar la superficie del receptor cubierta se ha modelado la presencia del ligando posicionando una sonda genérica de 1.95Å de radio en todos los puntos de la matriz cercanos a la superficie accesible a solvente del receptor. Esta superficie está definida por aquellos átomos que presentan un área de superficie accesible al solvente positiva (SASA>0, del inglés *Solvent Accessible Surface Area*). El valor de BSA es calculado como la diferencia de SASA al introducir una sonda. Como propiedad L se emplea el SASA del ligando. Además, la contribución a la desolvatación del ligando es también estimada empleando el potencial de desolvatación del ligando y el SASA del receptor. De este modo, la energia total de la interacción por desolvatación es obtenida por la suma de dos funciones de correlación, representando cada una la desolvatación del receptor al interaccionar con el ligando y viceversa.

Combinación de términos energéticos

Como se ha indicado, la aplicación FRODOCK emplea como criterio de ajuste la combinación del cálculo de cuatro correlaciones, de forma que la evaluación en el espacio rotacional para una determinada traslación se obtiene mediante:

$$C_{Rotacional} = FFT^{-1}\left[\sum_{l} d^{l}_{mh}(\pi/2)d^{l}_{hm'}(\pi/2)\left[\mathbf{w}_{\mathbf{vdw}}\mathbf{I}_{\mathbf{vdw}}{}^{l}_{mm'} + \mathbf{w}_{\mathbf{ele}}\mathbf{I}_{\mathbf{ele}}{}^{l}_{mm'} + \mathbf{w}^{\mathbf{recep}}_{\mathbf{desol}}\mathbf{I}^{\mathbf{recep}}_{\mathbf{desol}}{}^{l}_{nmm'} + \mathbf{w}^{\mathbf{ligand}}_{\mathbf{desol}}\mathbf{I}^{\mathbf{ligand}}_{\mathbf{desol}}{}^{l}_{nmm'}\right]\right] \quad (4.42)$$

Los valores w corresponden a la ponderación dada a cada interacción energética y determinan la importancia de cada una de ellas. Si bien la aplicación permite el establecimiento de estos valores por el usuario, valores por defecto se han establecido mediante un conjunto de entrenamiento (véase sección **5.2**). En general, puede observarse que el término de van der Waals es el término principal de la función de correlación, siendo la complementariedad de superficies que dicho término proporciona esencial para un correcto ajuste entre proteínas. Por su parte, el término de desolvatación y, en menor medida, el electroestático permiten corregir y refinar los posicionamientos obtenidos con el primer término.

4.2.2.2 Origen de las expansiones armónicas

Al igual que en el ajuste multi-resolución, la traslación entre las proteínas se obtiene mediante la variación de la representación radial armónica de una de ellas. Mientras que la proteína considerada fija mantiene el origen de su representación sobre su centro de masas, la considerada móvil varía la posición del origen de su representación. El establecimiento de la proteína más pequeña como fija permite obtener una mejor representación sobre las capas internas a la vez que reduce el número de capas necesario en el cálculo de la correlación. Como puede verse en la **Fig. 4.15**, manteniendo fija la estructura más pequeña (es decir, la estructura ligando) la zona de interacción entre las proteínas está representada en capas más cercanas al origen y por tanto mejor muestreadas. De este modo, la expansión sobre armónicos esféricos del ligando es realizada una sola vez sobre su centro. La traslación entre las proteínas se obtiene estableciendo como origen de la representación del receptor la posición relativa donde se quiere colocar el ligando. Además, las representaciones radiales armónicas

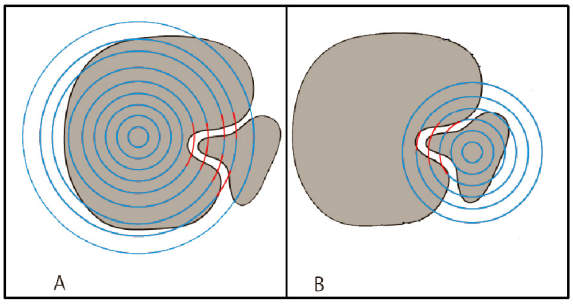

Fig. 4.15- Establecimiento del origen de las expansiones armónicas radiales sobre el receptor (A) y sobre el ligando (B) en ajuste externo. En el caso A, al ser el receptor el objeto más grande, la zona de interacción queda representada en capas alejadas del origen. En el caso B, al ser el ligando más pequeño, la interacción queda representada en capas más cercanas al origen. Esto proporciona una doble ventaja: el uso de capas internas permite una representación más detallada y simplifica el cálculo de correlación al disminuir el número de capas empleadas.

del receptor estarán limitadas a tantas capas como tenga la representación del ligando, reduciéndose de este modo el cálculo de la correlación al mínimo. Por tanto, en el cálculo de la integral (4.34) el término del ligando sólo debe ser calculado una única vez mientras que los coeficientes $\hat{f}_{lm}(\overline{r_j})$ asociados al receptor son calculados para cada traslación. Esto es así para todos los términos energéticos excepto para el término de desolvatación del ligando. En este caso los términos de la integral se invierten de forma que los coeficientes esféricos asociados al potencial de desolvatación del ligando son calculados una única vez y es la expansión de la accesibilidad del receptor la que varía para cada traslación.

4.2.2.3 Delimitación del espacio traslacional

El ajuste proteína-proteína es un ajuste exterior, de forma que una de las estructuras debe ser trasladada y rotada alrededor de la otra. De esta forma, los límites del espacio de búsqueda vendrán determinados por las dimensiones de las estructuras de ambas proteínas. La estimación de los puntos traslacionales validos se realiza mediante la construcción de una máscara que cubra completamente la estructura de la proteína más grande (la denominada receptor). Esta máscara es

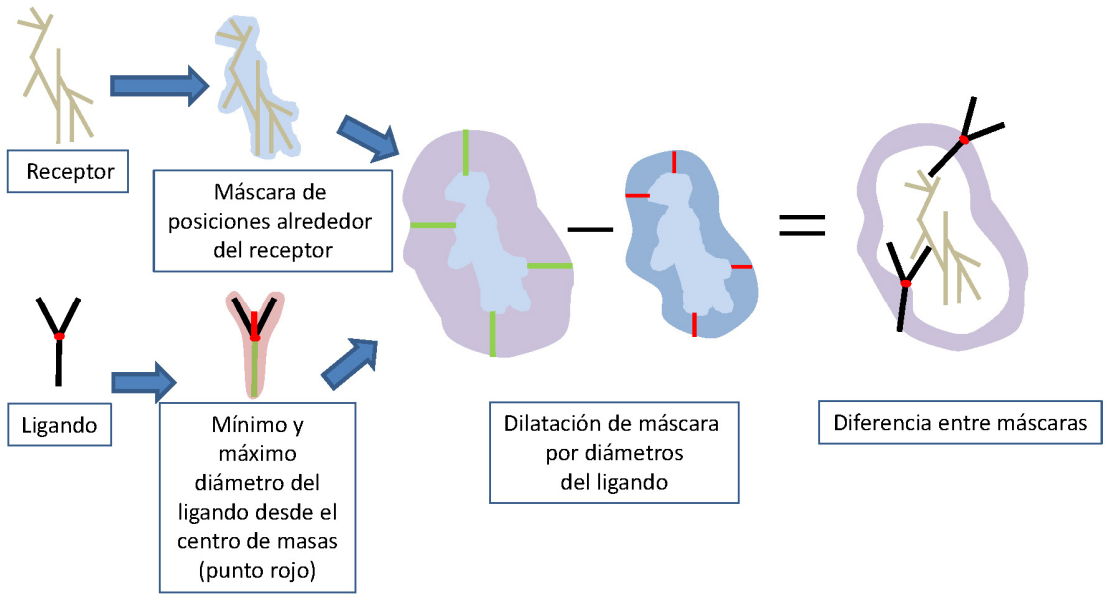

Fig.4.16- Esquema de determinación de posiciones traslacionales válidas entre receptor y ligando en ajuste externo. En función de las dimensiones del receptor y de los diámetros mínimo y máximo del ligando se determina el conjunto de posiciones traslacionales en las que puede situarse el ligando sin que esté demasiado próximo (en cuyo caso todas las rotaciones producirán colisiones) ni demasiado alejado (ninguna rotación permitirá contacto).

dilatada mediante dos distancias: el diámetro mínimo y máximo de la proteína más pequeña (ligando). El diámetro mínimo y máximo del ligando se establecen como la distancia del átomo más próximo y del átomo más alejado de la superficie al centro de dicha estructura. La diferencia entre las dos máscaras creadas de este modo excluye a posiciones excesivamente cercanas o lejanas. Las posiciones excesivamente cercanas producen profundas colisiones entre receptor y ligando para cualquier rotación de este último. Las posiciones excesivamente lejanas, por el contrario, no permiten establecer contacto para ninguna rotación (ver **Fig. 4.16**). Debido a que en las interacciones proteína-proteína no basta con un contacto mínimo entre las proteínas y que por otro lado las energías implicadas requieren cierta separación entre las proteínas, estos límites pueden ser reducidos mediante porcentajes correctamente adaptados a los casos de estudio.

4.2.2.4 Esquema general de funcionamiento de FRODOCK

A partir de todas las adaptaciones realizadas sobre el método FRM3 el esquema de funcionamiento de FRODOCK es mostrado en **Fig. 4.17**. Una descripción en mayor detalle de las características de la aplicación es proporcionada en [142].

4.3 Adaptación a computación Grid de aplicaciones bioinformáticas

Si bien la aplicación del método FRM3 en la resolución de los problemas de ajuste estructural descritos en secciones anteriores es efectivo y de gran utilidad para acelerar su realización, la complejidad de los problemas a tratar hace necesaria, en muchas situaciones prácticas, una gran capacidad computacional para resolverlos. Por este motivo, una paralelización de estas aplicaciones puede contemplarse como una alternativa para la resolución de estos problemas en márgenes de tiempo cortos. Esta paralelización puede ser realizada sobre cualquier tipo de entorno paralelo. Así, su implementación sobre *clusters* puede proporcionar un rendimiento adecuado. Sin embargo, la computación Grid y el amplio acceso a recursos computacionales que proporciona se ofrece como un marco ideal para ejecutar este tipo de aplicaciones de alta productividad, permitiendo una mayor adaptación del número de procesos paralelos adecuada al tamaño de las estructuras a ajustar. En las siguientes secciones se muestran las

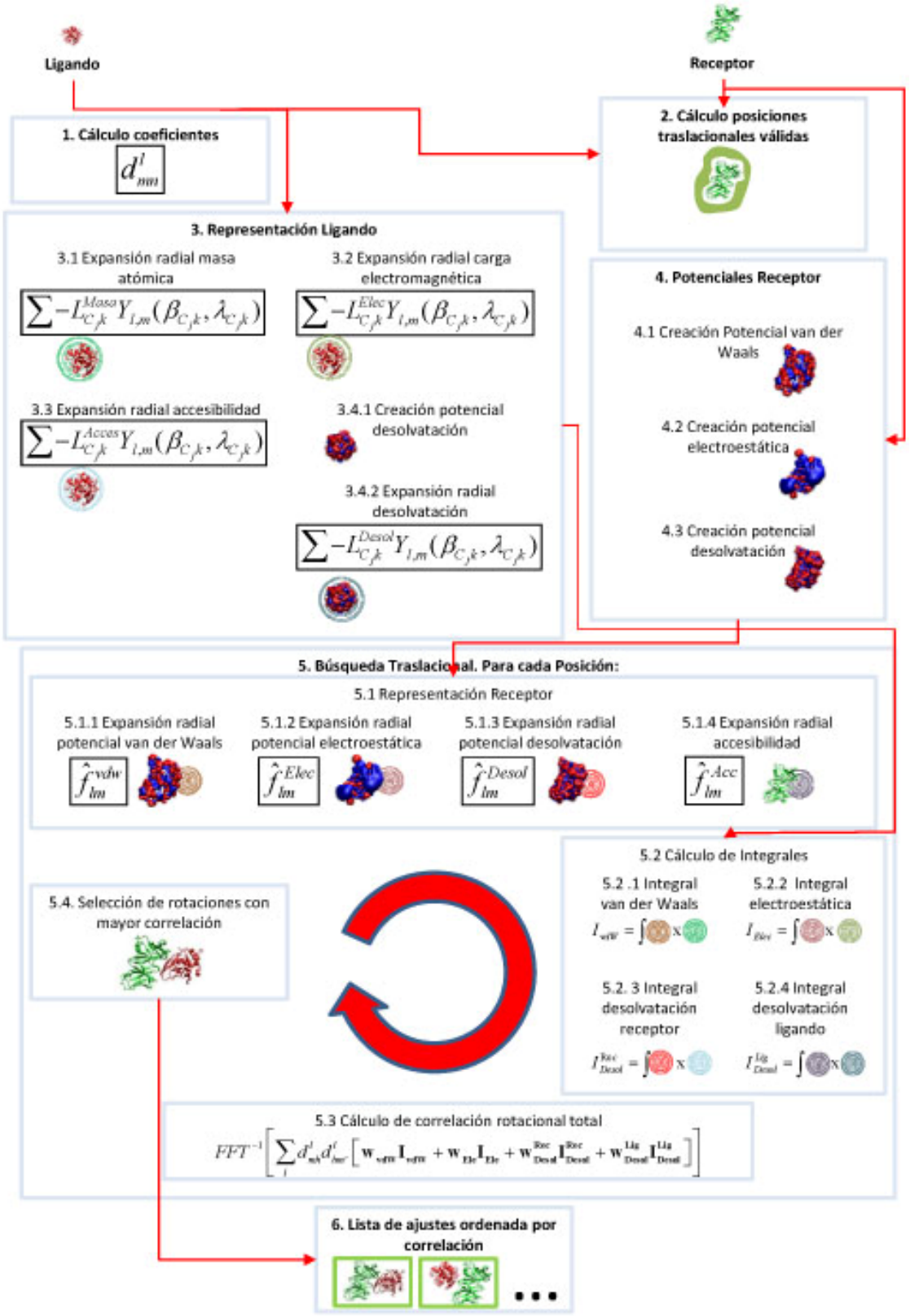

Fig.4.17- Esquema de funcionamiento de la aplicación FRODOCK. La secuencia en la que las acciones son realizadas es indicada por la numeración establecida. Las líneas rojas indican el flujo de datos entre las distintas acciones.

adaptaciones a computación Grid de las aplicaciones diseñadas en las secciones anteriores, mostrándose las motivaciones que las han orientado.

4.3.1 Aplicación ADP_EM

La eficiencia obtenida mediante FRM3 en la aplicación ADP_EM es muy elevada. Como puede observarse en los resultados obtenidos (véase sección **5.1**) los tiempos requeridos por esta aplicación en la predicción de la posición de una estructura atómica en un mapa de baja densidad requieren, en un computador estándar, tiempos de pocos minutos. Por tanto, una paralelización de esta aplicación no proporcionará una mejora importante. Sin embargo, una aproximación paralela puede ser beneficiosa cuando, sobre un mismo mapa a baja resolución, es necesario realizar un gran número de ajustes diferentes. Esta paralelización es de gran utilidad en la determinación de estructuras moleculares por homología. En este tipo de problemas se debe realizar el ajuste de un gran conjunto de modelos estructurales obtenidos por herramientas bioinformáticas de homología sobre un mismo mapa a baja resolución. Debido al gran número de métodos de predicción existentes, así como a la gran cantidad de predicciones que éstos realizan, el número de modelos que se dan en casos prácticos puede alcanzar el orden de varios miles. Una aproximación paralela puede realizar el ajuste de las diferentes estructuras de modo concurrente, determinando posteriormente qué estructuras presentan un mejor ajuste. Así, desde un punto de vista computacional, esta aproximación puede considerarse un proceso de alto rendimiento computacional donde la misma operación de ajuste es realizada, variando únicamente la estructura a ajustar en cada caso. Además, al emplearse en todos los casos el mismo mapa a baja resolución, es posible realizar los cálculos para determinar su representación armónica una sola vez en una etapa previa a la paralelización, reduciendo de este modo el cálculo del ajuste para procesos paralelos.

De esta forma, la adaptación paralela de ADP_EM a ejecución Grid [143] se realiza dividiendo el problema de determinar qué estructuras se ajustan mejor a un mapa a baja resolución en tres fases (**Fig. 4.18**):

- ***Fase de pre-computación***: Mediante una aplicación ejecutada de forma local se realiza la representación radial armónica del mapa a baja resolución. Esta representación tiene el origen sobre el propio centro del mapa. Además, a partir de las dimensiones del mapa se establecen los

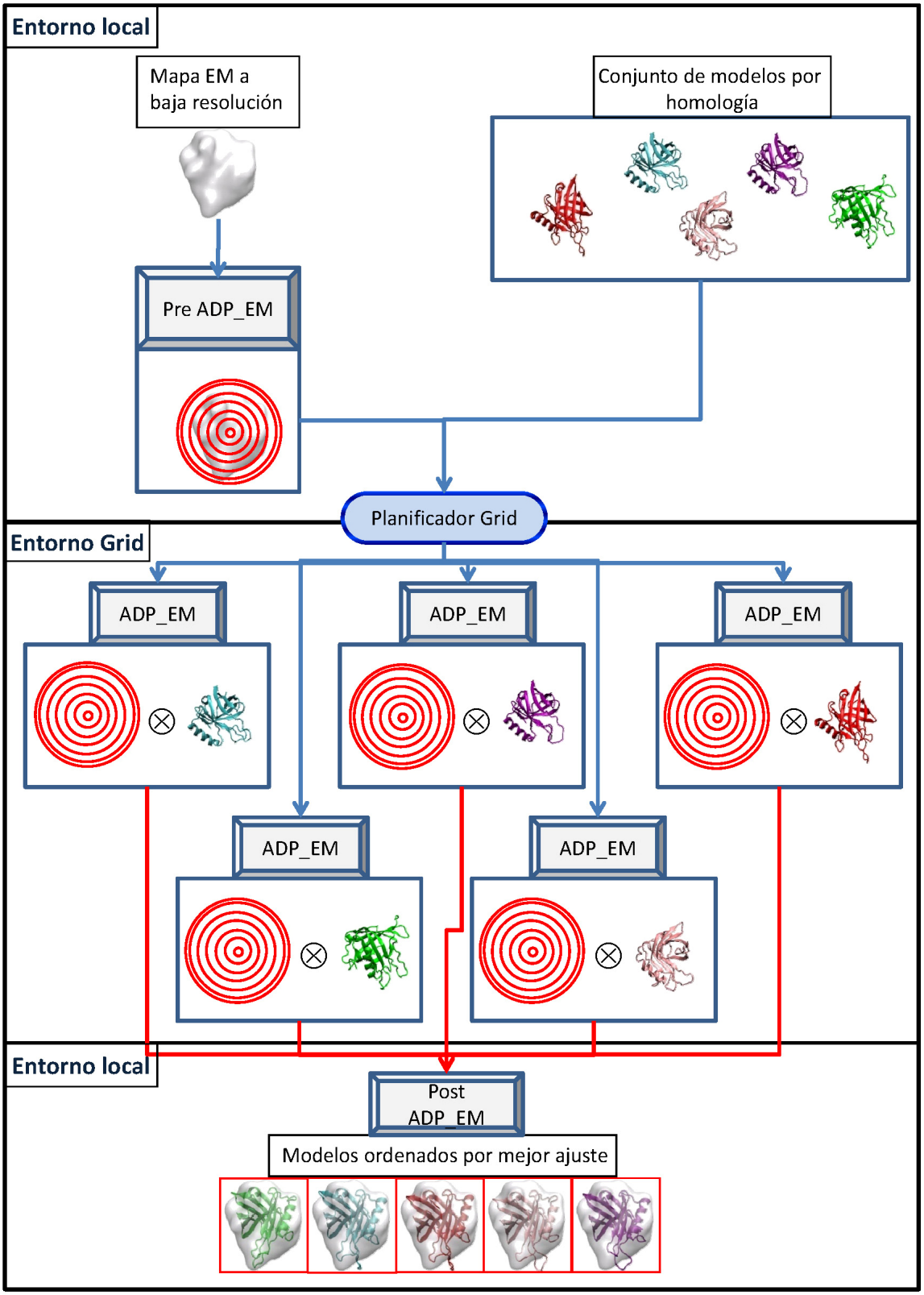

Fig.4.18- Realización de una evaluación de modelos por homología mediante ADP_EM en un entorno Grid. A diferencia de ADP_EM en ejecución no paralela, el mapa a baja resolución se mantiene fijo, de forma que su representación esférica se realiza una sola vez y puede ser empleada por todos los procesos paralelos. Un proceso preliminar realiza esta representación además de otras precomputaciones (coeficientes *d*, máscara traslacional). En el entorno Grid, el ajuste de cada modelo de homología se realiza sobre diferentes tareas paralelas. Finalmente, los ajustes realizados por todos las tareas paralelas se ordenan en función de la correlación. Los ajustes con mayor correlación determirán las estructuras homólogas más próximas a la real.

límites de búsqueda traslacional. Es importante destacar que, en este caso, el mapa a baja resolución será considerado fijo.

- ***Fase paralela de ajuste***: A través de un sistema de metaplanificación de trabajos sobre entornos Grid se realizará el lanzamiento de los diferentes procesos o tareas. A cada tarea se le proporcionará la representación armónica del mapa, los límites calculados para la búsqueda traslacional y una estructura diferente a ajustar. A partir de estos datos de entrada la aplicación ADP_EM se realiza siguiendo un esquema similar al de **Fig. 4.11**. La única diferencia importante en la metodología seguida en este caso concierne al objeto considerado fijo. Mientras que en la metodología descrita anteriormente se establece a la estructura como elemento fijo y por tanto su representación armónica es calculada una sola vez, en la adaptación paralela el elemento fijo es el mapa a baja resolución, realizándose la búsqueda traslacional mediante diferentes representaciones armónicas de la estructura. La ventaja de mantener la representación de la estructura atómica fija radicaba en que garantiza el cálculo de la correlación sobre las capas más interiores de las representaciones, con la mejora en la aproximación que ello conlleva. Sin embargo, en el caso de evaluación de estructuras homólogas, generalmente el tamaño de éstas será similar al del mapa al corresponderse con la estructura de toda la molécula. De este modo, desde el punto de vista de eficiencia en el cálculo de la correlación, carece de importancia qué representación se mantiene fija ya que una u otra determinan un mismo número de capas esféricas para realizar el cálculo de la correlación. Por ello, el mantenimiento del mapa a baja resolución como elemento fijo resulta más eficiente en este caso ya que permite aprovechar la pre-calculación de la fase anterior.
- ***Fase de combinación de resultados***: Una vez finalizadas todas las tareas en el entorno Grid, se comparan localmente las correlaciones obtenidas para todas las estructuras, estableciendo un orden entre ellas. Aquellas estructuras que presenten mayor correlación serán las que más se aproximen a la estructura real de la molécula estudiada.

4.3.2 Aplicación FRODOCK

A pesar de la drástica reducción de tiempo en la exploración de posibles ajustes que supone emplear el método FRM3, la realización de un ajuste mediante la aplicación FRODOCK puede llegar a requerir ejecuciones de horas e incluso días en un computador estándar. El uso de hasta tres términos energéticos de interacción en el cálculo de correlaciones y el hecho de que la exploración traslacional deba realizarse sobre un número de puntos que puede alcanzar el orden de 10^5 en función del tamaño de las estructuras son factores que implican unos importantes requerimientos computacionales. Además, el uso de herramientas de homología o el estudio de cambios conformacionales en las estructuras pueden proporcionar una gran cantidad de candidatos a ser ajustados.

Para afrontar esta gran demanda computacional la aplicación puede ser paralelizada con el fin de, empleando concurrentemente diferentes recursos computacionales, acelerar el tiempo de obtención de resultados. En este sentido, la naturaleza de la aplicación FRODOCK facilita su paralelización. Esta paralelización consistirá en la división del espacio de búsqueda, de forma que procesos concurrentes exploren diferentes regiones del espacio [119]. La forma más simple de dividir el espacio de búsqueda consiste en dividir el espacio de traslación (muestreado sistemáticamente en FRODOCK) en distintas regiones (ver **Fig. 4.19**). Mediante esta adaptación, la aplicación FRODOCK puede clasificarse como una aplicación de alto rendimiento computacional, caracterizándose por la ejecución de tareas independientes que realizan el mismo cálculo sobre un conjunto variable de parámetros de entrada. En el caso de FRODOCK, cada tarea paralela realizará el mismo tipo de cálculo (correlaciones entre términos energéticos) empleando diferentes representaciones radiales armónicas

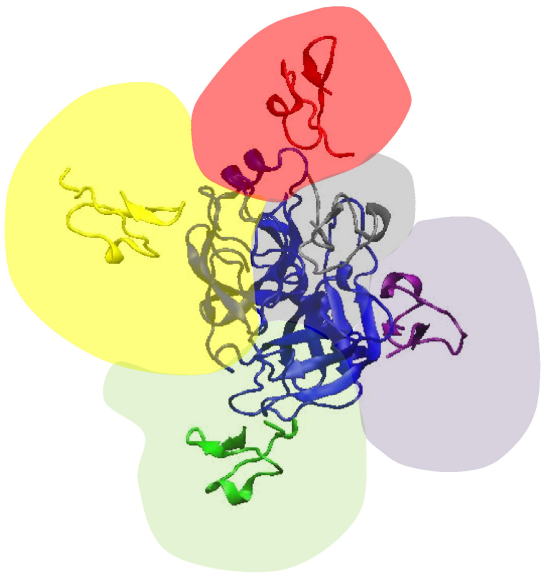

Fig.4.19- Partición del espacio de búsqueda traslacional en la paralelización de FRODOCK. Las diferentes áreas alrededor (zonas de color) del receptor representan subconjuntos de posiciones traslacionales. Los subconjuntos son independientes y pueden ser explorados por diferentes tareas paralelas.

del receptor dependientes de los posicionamientos del ligando.

Para la correcta adaptación de FRODOCK a un entorno de computación Grid se requiere tener presente sus detalles de implementación. Como ya se ha indicado anteriormente, tanto los coeficientes d^{l}_{mn} como los términos dependientes del ligando ($L_{C_jk}Y_{l,m}(\beta_{C_jk},\lambda_{C_jk})$) en la integral (4.34) son idénticos en el cálculo de correlaciones para todo punto traslacional que se explore, y por tanto pueden ser pre-calculados. Por ello, resultaría ineficiente que todos los procesos paralelos realizaran este cálculo compartido, resultando más adecuado realizarlo en un paso previo común a todos. Del mismo modo, una vez todos los procesos paralelos terminan su ejecución es necesario realizar una combinación de los resultados. Al finalizar, cada proceso devolverá un listado con posicionamientos del ligando ordenados por el valor de la correlación total. Así, será necesario reunir y reordenar todos los listados de posicionamientos en un único listado de soluciones. De esta forma, la aplicación es dividida en tres fases diferentes para su correcta realización en un entorno Grid (ver **Fig. 4.20**):

- ***Fase de pre-computación***: Aquellos datos comunes necesarios para todos los puntos de exploración traslacional son calculados por una aplicación pre-FRODOCK de forma local. Estos pre-cálculos consistirán principalmente en la creación de las representaciones armónicas del ligando. Además, esta aplicación también determinará el espacio de búsqueda traslacional que debe ser explorado (determinado a partir de las dimensiones del receptor y el ligando) y realizará la división de éste, creando diferentes conjuntos de puntos traslacionales para cada proceso paralelo que se realice.
- ***Fase paralela de ajuste***: Mediante un sistema de metaplanificación de trabajos sobre entornos Grid se realizará el lanzamiento de los diferentes procesos o tareas. A cada tarea se le deberá de proporcionar la información pertinente sobre el receptor, la información pre-calculada del ligando y un conjunto de puntos traslacionales a explorar.
- ***Fase de combinación de resultados***: Una vez finalizadas todas las tareas en el entorno Grid, se combinarán las listas de resultados obtenidas por éstas, reordenando la lista final en función de los valores obtenidos de correlación total. Esta operación se llevará a cabo mediante una aplicación local.

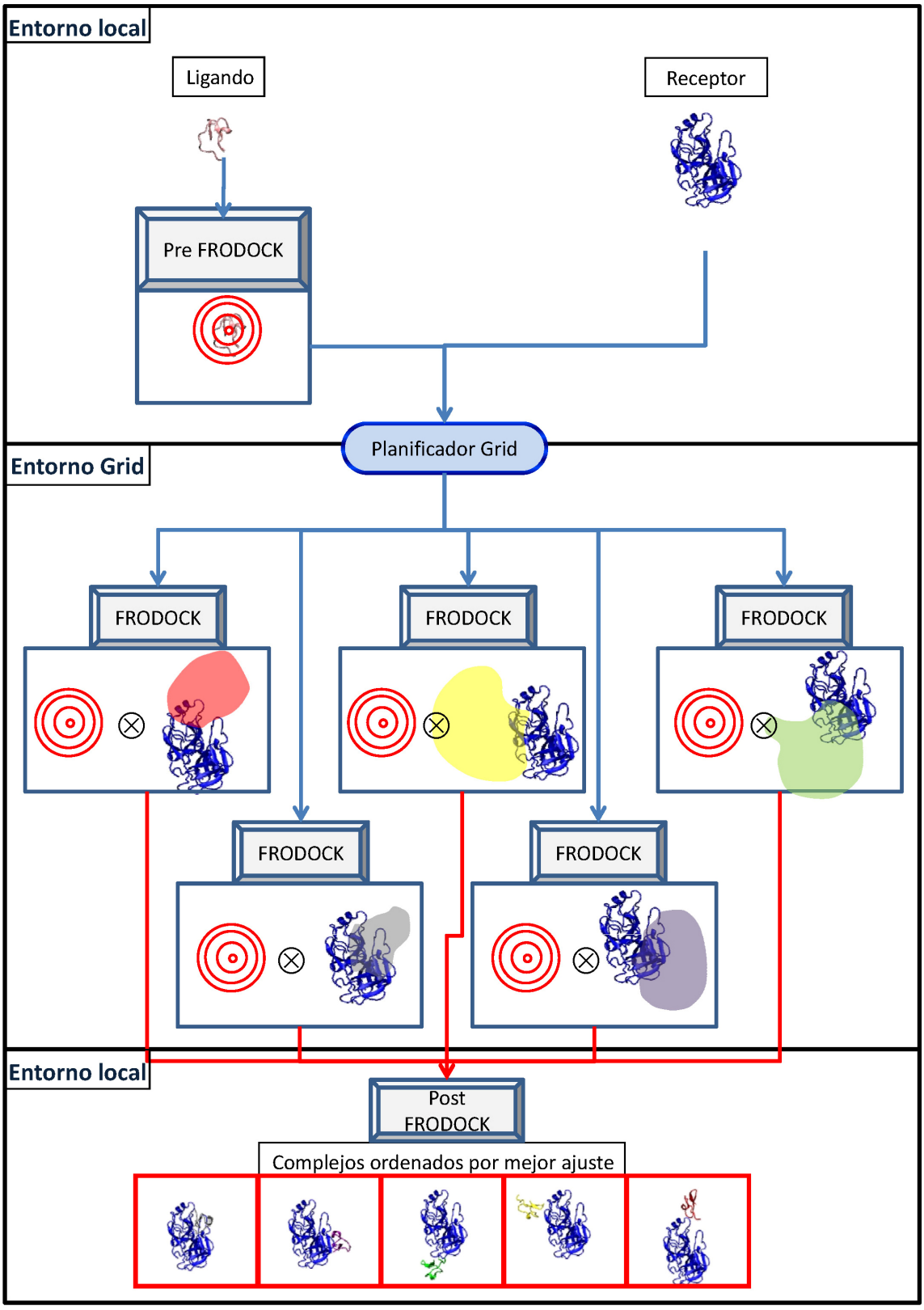

Fig.4.20- Realización de un ajuste proteína-proteína mediante FRODOCK en un entorno Grid. Un proceso preliminar realiza la representación esférica (fija) del ligando junto con otros precálculos (coeficientes *d*, máscara traslacional). Utilizando estas precomputaciones, diferentes procesos paralelos realizan la búsqueda de los mejores ajustes en diferentes zonas traslacionales. Finalmente, los ajustes realizados por todos los procesos paralelos son ordenados en función de la correlación. Los mejores ajustes determinan las conformaciones con más posibilidades de crear un complejo estable.

4.3.3 Aspectos de la paralelización en computación Grid

Como se ha indicado, la paralelización de este tipo de ajustes permite clasificarlos como aplicaciones de alto rendimiento computacional. Este tipo de aplicaciones precisan de la ejecución de un alto número de tareas, realizando cada una de ellas el mismo tipo de cálculo sobre distintos conjuntos de entrada. El amplio potencial computacional paralelo requerido para afrontar este tipo de problemas de manera rápida hace de los entornos Grid, que proporcionan una amplia cantidad de recursos computacionales, un marco adecuado para su realización. Sin embargo, las dos aplicaciones de ajuste presentadas muestran una caracterización más concreta. Así, si bien las aplicaciones realizan un mismo cálculo sobre diferentes conjuntos de datos de entrada, en ambos casos parte de este conjunto de datos es común a todas las tareas. En el caso de ADP_EM, el mapa a baja resolución sobre el que se ajustan los diferentes modelos es el mismo. En el caso de FRODOCK, todos los datos de entrada son idénticos para todas las tareas, variando únicamente las posiciones traslacionales relativas entre las proteínas. Este uso común de datos por parte de las tareas debe ser adecuadamente tratado por un sistema Grid con el fin de mejorar la productividad. En este sentido, el mantenimiento de funcionalidades cache sobre el sistema repercute de forma significativa sobre la productividad, ya que proporcionará una reducción de los tiempos de transferencia a través del sistema cuando más de una tarea sea ejecutada sobre un mismo recurso.

5 Validación de las aplicaciones de ajuste

Para determinar la fiabilidad de las aplicaciones ADP_EM y FRODOCK se han empleado conjuntos de pruebas estándar usados en el campo de la bioinformática estructural como marco de comparación. A grandes rasgos, los experimentos de validación tratan de modelizar una estructura atómica ya conocida a partir de información parcial o de distinta naturaleza. Dado que la estructura original se conoce, es posible evaluar el modelo reconstruido. Para ello, se calcula la distancia media entre las posiciones atómicas equivalentes (RMSD del inglés *Root Mean Square Deviation*) del modelo construido y la estructura original mediante:

$$RMSD(X,Y) = \sqrt{\frac{\sum_{i=1}^{n}(x_i - y_i)^2}{n}} \qquad (5.1)$$

Siendo X e Y los conjuntos equivalentes de n átomos de las estructuras reconstruidas y original. x_i e y_i corresponden a las coordenadas del átomo *i*-ésimo de cada conjunto. Esta medida de la bondad permite hacer comparaciones con otros métodos existentes, evaluándose así las aplicaciones desarrolladas. En función del tipo de problema y de su complejidad, se establecen diferentes límites en los valores de RMSD para determinar el grado de bondad de las soluciones.

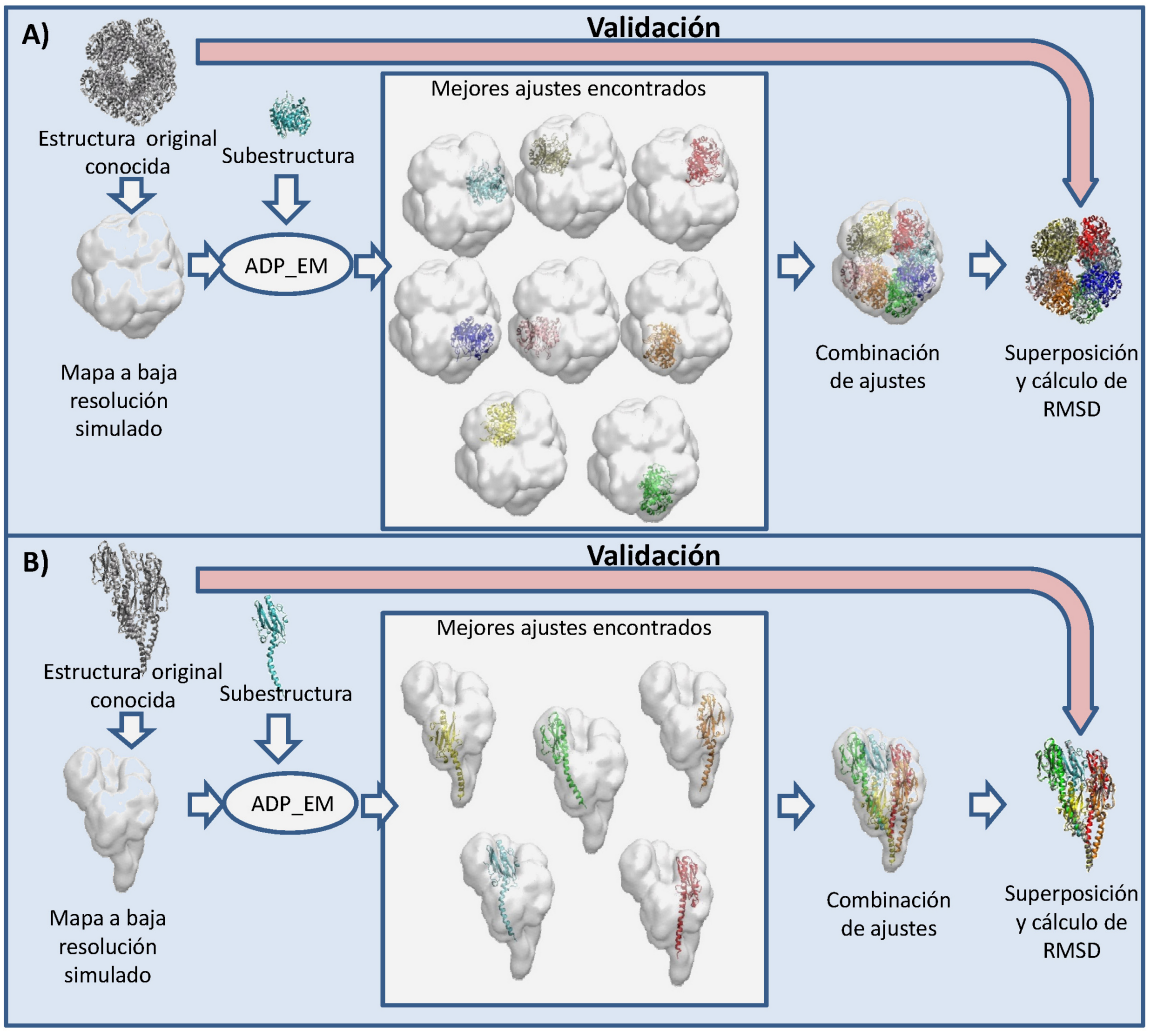

Fig.5.1- Ejemplos de ajuste multi-resolución con datos simulados a partir de la estructura conocida del complejo. (A) corresponde al complejo 5-aminolevulinato deshidrogenasa (PDB: 1aw5) formado por la combinación de ocho subestructuras idénticas. (B) corresponde a un complejo Pilin (PDB:2pil) formado por cinco subestructuras idénticas. En ambos casos, a partir de la estructura original del complejo se crea un mapa a baja resolución simulado. Empleando ADP_EM, las subestructuras son ajustadas sobre cada uno de los mapas. Los mejores ajustes proporcionan los diferentes posicionamientos de la subestructura, de modo que su combinación permite reconstruir la estructura del complejo. Finalmente, la estructura reconstruida se superpone a la original y se calcula el RMSD entre ambas. El valor de RMSD obtenido proporciona una medida de la calidad de los ajustes realizados y una medida de la validación.

5.1 Validación ADP_EM

Tal y como se ha indicado en el capítulo 4, la aplicación ADP_EM puede emplearse en la resolución de diferentes tipos de problemas o situaciones en los que es necesario el ajuste de una o varias estructuras atómicas en un mapa de baja resolución. Por tanto, se valorará la metodología propuesta en distintos escenarios. Para comprobar su validez en la caracterización de la estructura de complejos macromoleculares, la aplicación se analizará sistemáticamente empleando tanto mapas simulados a partir de estructuras reales como mapas y estructuras atómicas obtenidos experimentalmente. Además, para comprobar la

utilidad de la aplicación en conjunción con herramientas bioinformáticas de modelado, se empleará un conjunto de modelos de homología con el fin de determinar aquel que mejor aproxime la estructura real. En todos estos escenarios, los resultados serán comparados con aquéllos obtenidos mediante aplicaciones similares.

5.1.1 Validación sobre mapas simulados

Con el fin de determinar la capacidad de reconstruir complejos macromoleculares a partir de las estructuras atómicas de sus componentes se han empleado 28 casos simulados que incluyen complejos con una variada gama de tamaños y formas. En **Fig. 5.1** se muestran algunos de estos casos así como el esquema de validación empleado para evaluar la aplicación. Para cada caso, empleando la estructura atómica previamente conocida del complejo, se crearon mapas tridimensionales a diferentes resoluciones (10, 15, 20, 25 y 30Å) del complejo macromolecular. Estos mapas simulados fueron creados con un tamaño de celda o voxel de 3Å. Sobre cada uno de estos mapas del complejo se realizó el ajuste, independientemente, de las estructuras atómicas de sus componentes. Por completitud y para evitar situaciones de pre-alineamiento, el ajuste ha sido repetido 50 veces partiendo de diferentes posiciones aleatorias. Se han empleando diferentes distancias de muestreo rotacional (≈11º, 8º y 6º, asociadas a bases de esféricos armónicos de grado 16, 24 y 32 respectivamente) mientras que el muestreo traslacional se ha mantenido a 6Å (el doble del tamaño de celda de los mapas). Por último, para mejorar el contraste se ha empleado un filtrado Laplaciano.

Los resultados obtenidos han sido evaluados a partir del valor de RMSD de los componentes ajustados. Este valor se obtiene comparando la posición del componente ajustado con respecto a su posición en la estructura atómica original. De esta forma, se observa (ver **Fig. 5.2**) que incluso sobre mapas a muy baja resolución (30Å) en la mayoría de los casos se obtienen ajustes correctos con una precisión proporcional a la resolución del mapa. Como resultaba esperable, una menor resolución de los mapas del complejo dificulta determinar con precisión los posicionamientos correctos. Así, para resoluciones de 10 y 15Å se obtienen soluciones con desviaciones cercanas a 1Å de RMSD, lo cual puede considerarse como ajuste perfecto. Cuando la resolución de los mapas es de 30Å los ajustes obtenidos están cerca de los 3Å, siendo soluciones aceptables, mientras que

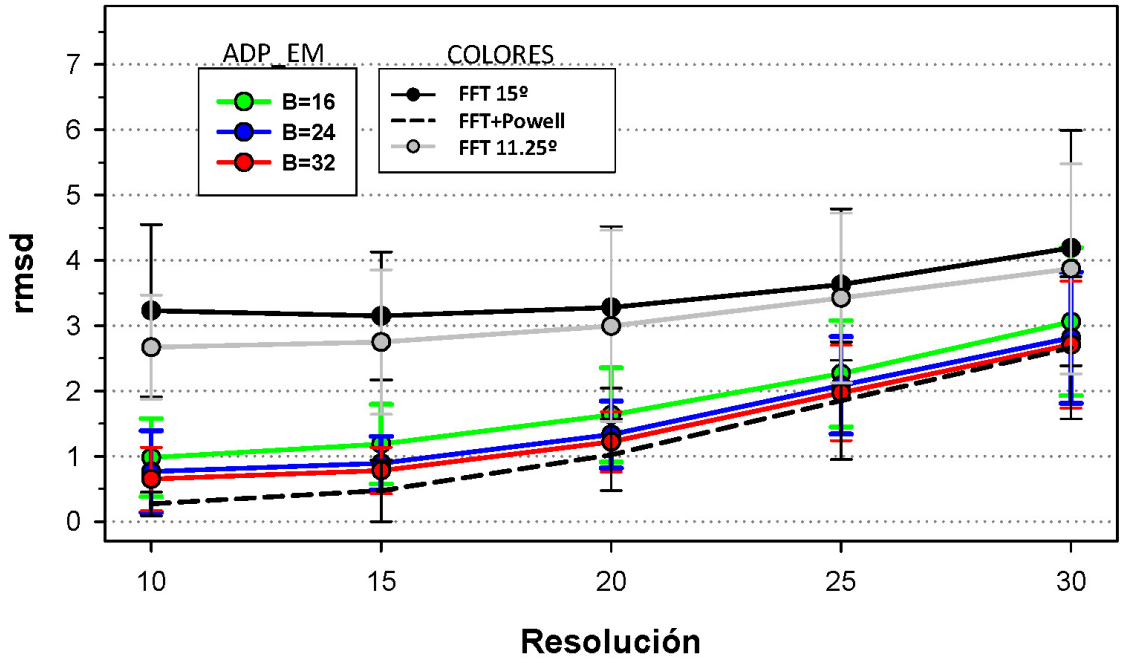

Fig.5.2- Precisión media del ajuste multi-resolución en el conjunto de casos simulados. Para determinar la precisión se calcula el RMSD entre la estructura del complejo original y la reconstrucción obtenida con los mejores ajustes. La precisión media ha sido calculada excluyendo aquellos resultados que no han producido una solución correcta (1aw5, 2pil, 1e6v y proteína ribosómica S2 con mapas a 30Å y GROES a partir de 15Å). El conjunto de 28 casos empleado se compone de los siguientes casos: 5-aminolevulinato deshidrogenasa (PDB: 1aw5,8x); cadenas A(2x), B(2x) y BC(2x) de metil-coenzima M reductasas (1e6v); glutamina sintetasa (1fpy,10x); ATP sulfurilasa (1g8g,6x); tricorn proteasa como homohexámero (1k32,6x); cadenas A(2x) y CD(2x) de quinol-fumarato reductasa (1kf6); ATP sulfurilasa (1g8g,6x), halo-gliceraldehido-3-fosfato deshidrogenasa (1gd1,4x); glutamato deshidrogenasa (1l1f,6x); tricorn proteasa como homotrímero (1n6d,3x); Cu-nitrato reductasa (1nic,3x); anillo de proteasoma (1j2p,7x); cadherin (1q5b,3x); lectin (1w3a,6x); hemaglutinina (1ruz,3x) RecA (1xmv,6x); cadena A de subunidad-ß2 canal de potasio dependiente de voltaje (2a79,4x); pilin (2pil,5x); catalasa (7cat,4x); GroEL-GroEs (1aon,7x); termosoma (1a6d,14x); subunidades grande y pequeña de ribosoma (1ffk + 1fjf); y proteína ribosómica S2 (1ffk – 1fjf). Para prevenir situaciones de pre-alineamiento, los ajustes con ADP_EM han sido repetidos 50 veces partiendo de posicionamiento aleatorios de las estructuras atómicas. Todos los casos han sido realizados con ADP_EM empleando tres distancias de muestreo rotacional diferentes: 11°, 8° y ~6°. El muestreo translacional se ha mantenido fijo a 6Å. En el caso de COLORES, los ajustes han sido repetidos diez veces variando aleatoriamente el posicionamiento de las estructuras. Esta aplicación ha sido empleada sin optimización (opción `-nopowell`) y con muestreos rotacionales de 15° y 11.25°. El resto de parámetros han sido mantenidos por defecto. La optimización (línea punteada) ha sido realizada a partir de los resultados obtenidos con el muestreo rotacional a 15° mediante la minimización de Powell.

únicamente en cinco casos (*5-aminolaevulinato deshidrogenasa*, *pilin*, *metil-coenzima M reductasa*, proteína ribosómica *S2* y *GroEs*) la falta de resolución impide encontrar ajustes aceptables. Además, el caso *GroEs* es el único para el que no ha sido posible encontrar una solución para resoluciones medias o bajas (15Å o más). Este caso ya había sido identificado como difícil por otros autores [144]. En cuanto al muestreo rotacional empleado, en **Fig. 5.2** se aprecia que al incrementarse el grado de la base de armónicos esféricos empleados (*B*), y por tanto siendo el muestreo más fino, las soluciones obtenidas presentan una mayor precisión en el ajuste. Sin embargo, la mejora obtenida de un muestreo a otro es poco significativa, por lo que será preferible un valor *B* pequeño (16, proporcionando un intervalo de muestreo de 11º) que, si bien devuelve soluciones

ligeramente peores, se mantiene dentro del margen de error esperable y conlleva un tiempo de ejecución significativamente más corto.

Como puede observarse, los valores de RMSD para las soluciones obtenidas en la mayoría de los casos se encuentran por debajo de 1/10 de la resolución de los mapas empleados. Esta relación establece el mínimo RMSD que se puede obtener con mapas simulados [145], aunque otros autores elevan esta relación a 4/10 en condiciones reales con mapas experimentales [131]. Por tanto, la precisión de la aplicación obtenida en este tipo de ajustes se presenta como muy alta.

Por último, en la misma **Fig. 5.2** se comparan los resultados con aquellos obtenidos con la herramienta de ajuste COLORES [145], integrada en el conjunto de programas Situs [92]. Esta aplicación de ajuste es una de las más rápidas y fiables de las actualmente existentes, siendo la herramienta estándar en el campo del ajuste multi-resolución. COLORES realiza la búsqueda del mejor ajuste mediante el uso del método FTM (véase sección **4.1**), de forma que acelera la búsqueda en el espacio traslacional. Los resultados mostrados han sido obtenidos empleando COLORES con una distancia de muestreo rotacional de 15º y 11.25º. Como puede observarse, los valores de RMSD obtenidos mediante ADP_EM son significativamente mejores que los obtenidos por COLORES. Únicamente realizando una optimización posterior de los mejores resultados obtenidos por COLORES mediante el método de Powell pueden conseguirse mejores soluciones. Sin embargo, esta optimización supone un incremento sustancial del tiempo de ejecución. Aun sin este incremento del tiempo, la aplicación COLORES requiere un tiempo medio para la realización del ajuste en los ejemplos empleados de aproximadamente 20-30 minutos en un computador estándar (Linux a 2.2MHz). Por el contrario, en el mismo soporte, ADP_EM requiere de tiempos de entre 4 minutos y pocos segundos en función del tamaño de muestreo empleado (ver **Tabla 5.1**). Por otro lado, los tiempos de ejecución de ADP_EM también han sido comparados con la aplicación 3SOM, considerada como la aplicación de ajuste más rápida de las existentes [144]. Esta aplicación realiza la maximización de complementariedad de superficies mediante superposiciones basadas en transformaciones vectoriales. Sin embargo, la comparación de la bondad de los resultados entre ADP_EM y 3SOM no ha sido realizada debido a que esta última aplicación proporciona una gran cantidad soluciones, de forma que la diferenciación de la solución correcta con respecto de aquellas espurias debe realizarse mediante una inspección visual. Aun con esta limitación, los tiempos de

	B/grados	10Å	15Å	20Å	25Å	30Å	Media	Ratio
ADP_EM	16/11º	28	31	35	34	38	34	x1
	24/8º	100	108	119	118	123	113	x3
	32/6º	226	220	225	216	221	222	x6
Búsqueda FFT	-/15º	1697	1926	2341	5028	6681	3543	x104
Minim. Powell	-/15º	375	918	1747	3739	6597	2675	x78
COLORES (FFT+Min.Powell)	-/15º	2072	2844	4088	8767	13278	6209	x182

Tabla 5.1- Tiempos medios, en segundos, obtenidos para ajuste multi-resolución del conjunto de casos simulados. Todas las ejecuciones han sido realizadas en un PC Linux con un procesador Xeon a 2.8GHz. El programa COLORES (versión 2.2.1) del paquete Situs fue empleado con un muestreo rotacional de 15°. Para la ejecución sin optimización se empleó la opción "–nopowell". El tiempo empleado por COLORES es igual a la suma de la búsqueda FFT simple y la minimización mediante Powell. En todos los casos la desviación estándar es del orden de magnitud de la media. La correlación en ambas aplicaciones fue realizada empleando un filtro Laplaciano para incrementar el contraste.

ejecución han sido comparados poniendo de manifiesto la mejor escalabilidad de ADP_EM al aumentar el tamaño de los mapas empleados. Así, en el caso del componente más grande de la *ribosomal protein S2*, 3SOM requiere prácticamente una hora de cálculo frente a los pocos minutos de ADP_EM. 3SOM es más eficaz únicamente cuando el ajuste se realiza sobre mapas tridimensionales de muy pequeñas dimensiones, que a efectos prácticos son muy difíciles (incluso imposibles) de caracterizar por EM.

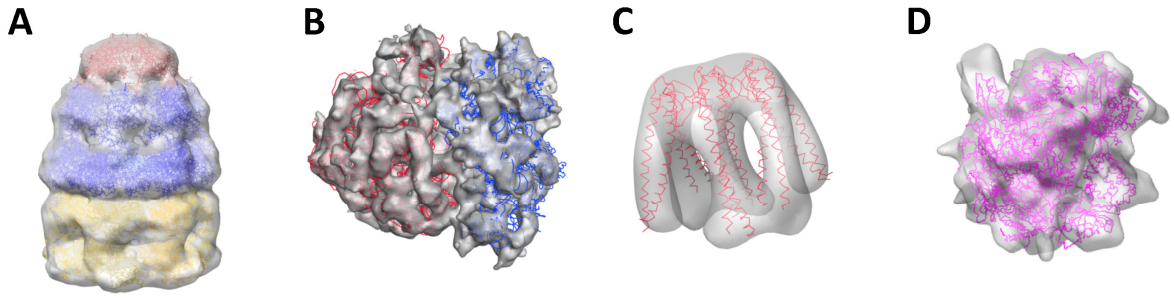

Fig. 5.3- Resultados de ajuste sobre mapas a baja resolución obtenidos experimentalmente por EM. **A)** ajuste de subunidades sobre mapa de GroES-ADP7-GroEL-ATP7 de *E.coli* a 23.5 Å (EMD ID 1046, PDB: 1ml5); en este caso las subunidades ADP y ATP de GroEL han sido ajustadas independientemente para reconstruir los anillos heptaméricos del complejo. En el caso de GroES se ha utilizado el heptámero completo. **B)** Ajuste de subunidades 30S y 50S en el mapa del ribosoma de *E.coli* a 14 Å (EMD ID 1046, PDB: 1gix/1giy). **C)** Ajuste de estructura atómica completa sobre el mapa de prefoldina a 23 Å ([146], PDB: 1l6h). **D)** Ajuste de estructura completa sobre el mapa RNA polimerasa II de levadura a 15 Å ([147], PDB: 1fxk).

5.1.2 Validación sobre mapas experimentales

Debido a que no existe un conjunto amplio de datos experimentales obtenidos por EM cuyas estructuras atómicas estén también caracterizadas, como conjunto de validación han sido empleados cuatro mapas contrastados previamente por otras aplicaciones o proporcionados por laboratorios afines. La **Fig. 5.3** presenta los mapas a baja resolución empleados y los correspondientes ajustes obtenidos mediante ADP_EM. Mientras que para GroES-ADP7-GroEL-ATP7 (A) y el ribosoma de *E.coli* (B) han sido ajustados los componentes que conforman el complejo, para la Prefoldina (C) y la levadura *RNA* Polimerasa (D) el ajuste se ha realizado empleando la estructura atómica completa. En todos los casos, empleándose un muestreo rotacional de 11º y un muestreo traslacional de 3Å, se obtienen resultados con un ajuste correcto, en consonancia con ajustes realizados con otras herramientas de ajuste. Así, por ejemplo, el resultado obtenido para la levadura *RNA* Polimerasa reproduce los resultados obtenidos mediante un laborioso ajuste manual en [148]. Únicamente en el caso (A) los componentes *GroES* (en rojo) no han podido ser correctamente ajustados debido a su pequeño tamaño y a la baja resolución del mapa empleado. Para obtener un ajuste correcto fue necesario emplear el heptámero completo (las siete estructuras *GroES* juntas). La bondad de los resultados obtenidos es totalmente comparable a los resultados proporcionados por la aplicación COLORES. Sin embargo, los tiempos de ejecución de ADP_EM, sobre todo para las estructuras y mapas más grandes, resultan claramente más competitivos, reduciéndose entre uno y dos órdenes de magnitud los tiempos de ejecución. Por ejemplo, un adecuado ajuste mediante COLORES del componente *GroEL* sobre el mapa experimental (A) empleando COLORES requiere una hora de cálculo frente a los 40 segundos de ADP_EM. En el caso de *GroES*, las diferencias de tiempo fueron de 11 horas respecto a 296 segundos para el componente más grande. En cuanto a los tiempos obtenidos con 3SOM, en **Tabla 5.2** puede observarse la mejora proporcionada por ADP_EM. Es importante resaltar que la reducción en tiempo obtenida no sólo se debe a la aceleración conseguida sobre la búsqueda en el espacio rotacional mediante FRM3. También juega un papel importante la ajustada acotación sobre la exploración del espacio traslacional que se consigue al ser muestreada de forma sistemática. Esta acotación resulta muy eficiente en casos como el de la Prefoldina, donde existen zonas huecas en partes centrales del mapa. En aplicaciones como COLORES, que emplean el método FTM, la

	GroEL-GroES	GroEl-ATP	30S	50S	RNAP	Prefoldina
3SOM	39	97	1320	3840	420	780
ADP_EM	38	98	223	295	20	6
Aceleración	1	1	6	13	21	130

Tabla 5.2- Tiempos medios (en segundos) empleados en el ajuste multi-resolución sobre mapas experimentales con ADP_EM y 3SOM. Todos los tiempos han sido obtenidos en una computadora PC con procesador Xeon a 2.8 GHz. La tercera fila muestra el factor de aceleración obtenido mediante ADP_EM con respecto a 3SOM.

acotación no puede realizarse, de forma que dentro de la búsqueda se exploran traslaciones sin sentido físico. En resumen, a la vista de la comparación de resultados entre ADP_EM y COLORES, resulta significativa la mejora que supone la aceleración de la búsqueda en el espacio rotacional (FRM) frente a la aceleración del espacio traslacional (FTM).

5.1.3 Ajuste 6D con modelos por homología

No siempre se puede disponer de las estructuras atómicas de componentes para la reconstrucción de la estructura de un complejo o una molécula visualizada mediante EM. En tal caso, una alternativa para determinar la estructura consiste en la utilización de herramientas bioinformáticas que, a partir de las estructuras de otras proteínas con secuencias de aminoácidos parecidas, producen modelos cercanos a la estructura objetivo. Este procedimiento se conoce como modelado por homología y constituye uno de los campos con mayor proyección en bioinformática [149]. Sin embargo, el número de los modelos realizados por homología puede ser muy alto con el fin de cubrir todas las posibles variantes conformacionales. Este número es aún mayor si se considera las distintas aproximaciones existentes en modelado molecular. En este sentido, un ajuste multi-resolución puede ser empleado desde dos puntos de vista diferentes:

- Desde la perspectiva de la reconstrucción de estructuras, el ajuste de los diferentes modelos permitirá interpretar mapas a baja resolución cuya estructura se desconoce [150].
- Desde el punto de vista de la reconstrucción por homología, el ajuste puede emplearse para evaluar la capacidad de una determinada herramienta de homología a la hora de modelar la estructura estudiada.

Para determinar la conveniencia de la aplicación ADP_EM en este tipo de problemas, se ha empleado un banco de pruebas [151] desarrollado a este efecto [152]. Este banco de pruebas está formado por ocho casos, cada uno de los

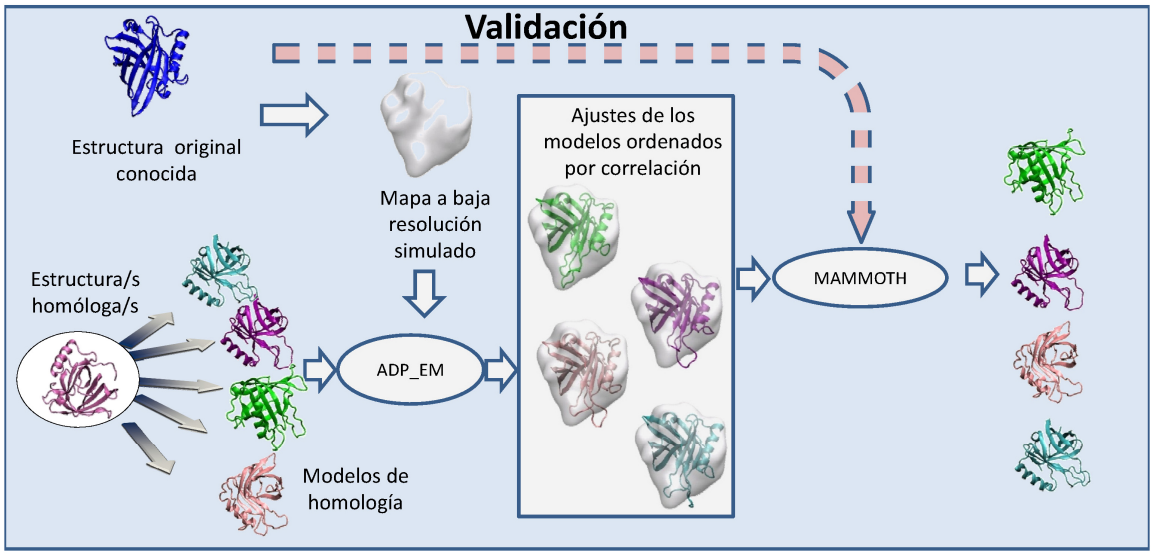

Fig.5.4- Esquema de validación del ajuste multi-resolución de modelos homólogos. Los modelos creados a partir de estructuras homólogas son ajustados a un mapa a baja resolución simulado de la estructura objetivo. El ajuste permite ordenar los modelos en función del grado de ajuste o correlación. Este ordenamiento debe ser validado *a posteriori* por el alineamiento de los modelos con la estructura original mediante MAMMOTH.

cuales se compone de un par de estructuras atómicas y varios mapas a baja resolución. Una de las estructuras atómicas corresponde a la estructura objetivo, mientras que la otra es una estructura homóloga (presentando entre un 12 y un 32% de identidad en la secuencia de aminoácidos con la estructura objetivo) que ha sido empleada para crear los modelos homólogos. Los mapas a baja resolución son simulados a distinta resolución a partir de la estructura objetivo. A partir de la estructura homóloga han sido creados 300 modelos mediante la herramienta MODELLER [132]. El objetivo del sistema de validación es determinar el mejor modelo mediante su ajuste a los mapas. Para validar numéricamente los resultados obtenidos se ha empleado el programa MAMMOTH (*Matching Molecular Models Obtained from Theory*) [153] que permite realizar alineamiento estructural entre los modelos y la estructura objetivo y cualificar el grado de similitud entre estructuras (en una medida equivalente a la obtenida por RMSD). El esquema del procedimiento de validación se muestra en **Fig. 5.4**. Los resultados obtenidos con todos los modelos validan su uso en este tipo de problemas. Como ejemplo ilustrativo, en **Fig. 5.5** se muestra la relación entre la correlación de ajuste obtenida con ADP_EM (empleando un muestreo traslacional de 1Å y rotacional de 11º) y el valor de alineamiento estructural proporcionado por MAMMOTH para todos los modelos creados para la proteína *1MUP*, incluyendo además la propia estructura objetivo y la estructura homóloga. Para realizar el ajuste de las estructuras se ha empleado un mapa a 12Å de resolución con una tasa de ruido

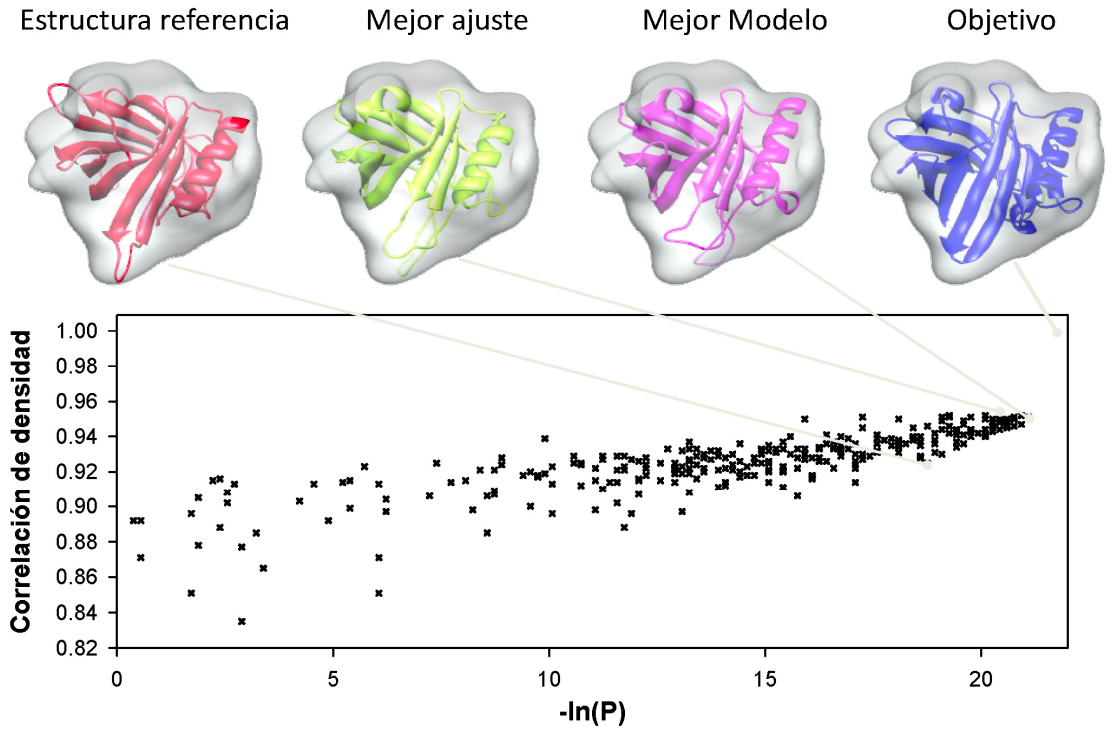

Fig.5.5- Relación entre correlación, obtenida por ADP_EM, y alineamiento estructural, obtenido por MAMMOTH, en el caso *1MUP*. El ajuste de las estructuras de los modelos por homología con ADP_EM se ha realizado sobre un mapa a baja resolución simulado a partir de la estructura objetivo. Se ha empleado un muestreo rotacional de 11º. El alineamiento estructural de los modelos se ha realizado con la estructura objetivo conocida. El valor de alineamiento está definido por –ln(P), siendo P la probabilidad de encontrar al azar una determinada proporción de aminoácidos alineados con respecto a la estructura más pequeña. La estructura referencia se corresponde a la estructura de la molécula a partir de la cual se han construido los 300 modelos. El mejor modelo es aquel que muestra mejor alineamiento, siendo al mismo tiempo uno de los mejores ajustados (octavo mejor modelo ajustado). Igualmente el modelo con mejor ajuste presenta un buen alineamiento, siendo casi insignificantes las diferencias entre éste y el mejor modelo.

artificial del 25%. Como puede observarse, es significativa la correlación existente entre ambos valores, de forma que los modelos mejor ajustados presentan también un alto alineamiento estructural y por tanto son más parecidos a la estructura objetivo. En concreto, el modelo que mejor se alinea a la estructura objetivo se encuentra entre los diez modelos con mejor ajuste. Además, como era esperable, la propia estructura objetivo obtiene el ajuste más alto. Respecto a la eficiencia obtenida mediante ADP_EM, el ajuste de un modelo puede necesitar desde menos de medio minuto (muestreo de 11º, B=16) a 1-2 minutos (6º, B=32) en función del muestreo rotacional empleado. Si bien un muestreo bajo de 11º permite discriminar los modelos más cercanos al objetivo, un muestreo más exhaustivo (6º) permite un mejor ajuste y por tanto, una mejor discriminación. Otros autores han realizado comparaciones entre diferentes aplicaciones en este tipo de problemas. Así, se han comparado los resultados obtenidos con FOLDHUNTER [154], que emplea entre 10 y 15 minutos por modelo, con los

devueltos por un protocolo optimizado por Monte Carlo (Mod_EM), el cual ajusta un modelo en 1-2 minutos. Si bien los resultados de ajuste son comparables en los tres casos, ADP_EM es considerablemente más rápido que FOLDHUNTER y comparable a Mod_EM. Sin embargo, a la vista de los resultados observados en [152] es esperable que para casos reales con mapas más grandes, la eficiencia de Mod_EM se reduzca considerablemente. Es muy importante resaltar la gran importancia que tiene la obtención de tiempos eficientes en la realización de este tipo de ajustes. Tiempos del orden de la decena de minutos en la realización de un ajuste son del todo inasumibles teniendo en cuenta que, en condiciones experimentales reales, el número de modelos por homología puede alcanzar fácilmente el orden de millares. Por ello, métodos rápidos como ADP_EM resultan adecuados para afrontar este tipo de problemas de alta productividad computacional.

5.2 Validación FRODOCK

Para determinar la validez de la aplicación de ajuste proteína-proteína se han empleado 76 complejos receptor-ligando obtenidos del conjunto de pruebas establecido como estándar para la evaluación de este tipo de herramientas de ajuste [155]. Cada caso de este conjunto presenta cuatro estructuras: para cada una de las dos proteínas que interaccionan (denominadas receptor y ligando) se proporciona su estructura tanto en estado libre como su conformación al estar unida a la otra proteína. Las estructuras libres son utilizadas como entrada de la aplicación de ajuste mientras que las estructuras unidas del complejo se emplean para validar *a posteriori* las conformaciones resultantes. El esquema de validación empleado se describe en **Fig. 5.6**. Para determinar la bondad de un ajuste pueden ser empleadas dos medidas diferentes de RMSD en función de la disposición relativa del ligando y de la región de interacción respectivamente:

- ***RMSD de ligando ($RMSD_L$)***: se realiza el alineamiento de las estructuras del receptor (del complejo solución conocido y del complejo obtenido por ajuste) y el posterior cálculo del RMSD entre las estructuras del ligando.
- ***RMSD de interfaz ($RMSD_I$)***: se determinan aquellos aminoácidos tanto del receptor como del ligando que están presentes en la interfaz de interacción en el complejo objetivo. Un aminoácido del receptor se considera en la interfaz si alguno de sus átomos está a una distancia menor

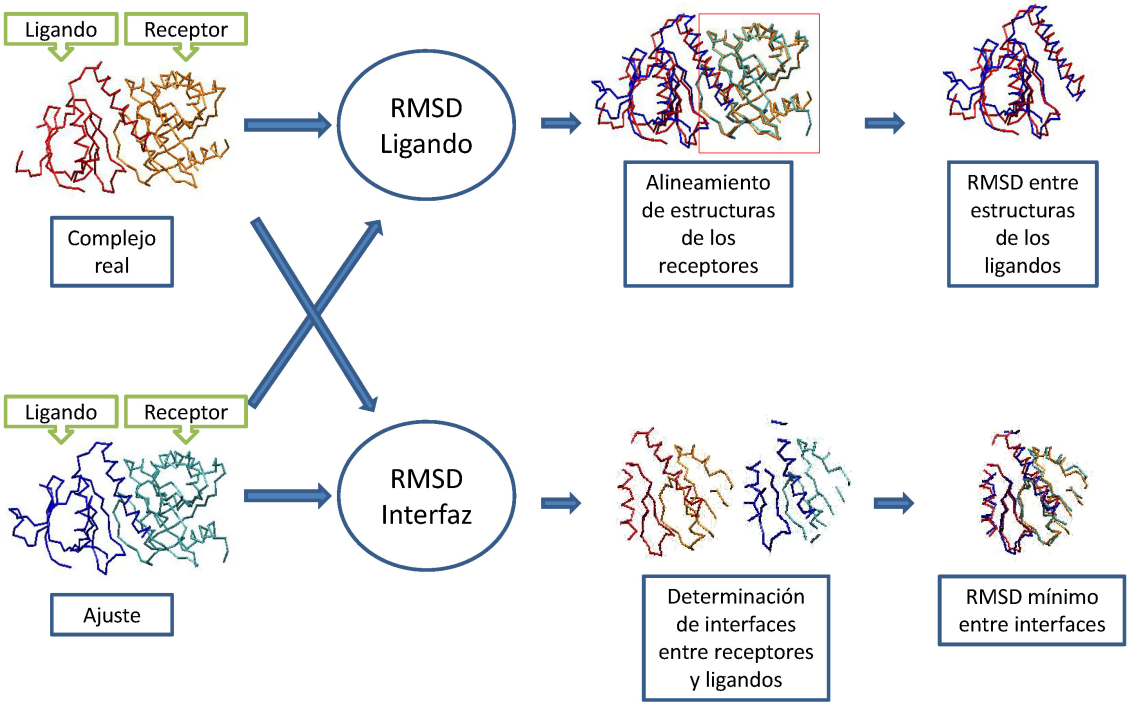

Fig.5.6- Evaluación de un ajuste proteína-proteína mediante $RMSD_L$ y $RMSD_I$. Para el cálculo del RMSD ligando, en primer lugar se alinean las estructuras del receptor del complejo real (en naranja) y del ajuste (azul claro), posteriormente se calcula el RMSD de las estructuras ligando (en rojo y azul oscuro respectivamente). En RMSD interfaz se determinan los aminoácidos que forman la interfaz de contacto receptor/ligando en el complejo real. Estos aminoácidos de la interfaz se alinean y se calcula su RMSD.

de 10Å de cualquier átomo del ligando. Los aminoácidos del ligando en la interfaz se determinan de forma similar. Una vez definidos los aminoácidos que conforman la interfaz, se realiza su alineamiento con los aminoácidos equivalentes en el complejo obtenido en el ajuste. El RMSD minimizado para estos aminoácidos es el valor $RMSD_I$ empleado para su evaluación.

Para una descripción más detallada de la evaluación de complejos proteína-proteína véase [156]. Una solución se considera aceptable si presenta valores por debajo de 10Å para el $RMSD_L$, siendo de calidad media si este valor no supera los 5Å. En el caso de emplearse $RMSD_I$, los valores que determinan las soluciones aceptables y medias son 4Å y 2.5Å respectivamente.

De los 84 casos presentes en el conjunto, aquellos que han sido clasificados como muy difíciles no han sido estudiados. En estos casos la estructura de las proteínas muestra importantes cambios conformacionales en su paso del estado libre a la conformación del complejo. Por ello, un ajuste como FRODOCK, que no proporciona flexibilidad, no permitirá obtener predicciones adecuadas. Los 76 casos restantes se agrupan en tres categorías: interacciones enzima-substrato (subconjunto E, 23 casos), interacciones anticuerpo-antígeno (A, 21 casos) y otros (O, 32 casos).

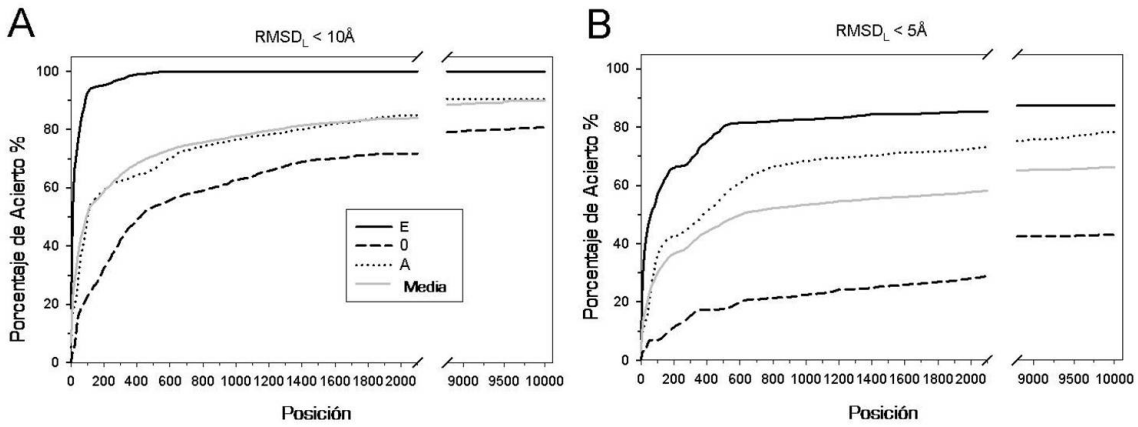

Fig.5.7- Porcentaje de acierto obtenido, evaluando mediante $RMSD_L$, en el conjunto de pruebas empleado para evaluar FRODOCK. La línea continua representa el porcentaje en casos Enzima-Substrato, la línea punteada corresponde a casos anticuerpo-antígeno, la línea discontinua a los casos de tipo O y la línea gris al porcentaje medio. El porcentaje de acierto es definido como el porcentaje de casos en los que se encuentra una solución aceptable o buena por debajo de la posición indicada en la abscisa. Las gráficas **A)** y **B)** muestran los porcentajes para soluciones aceptables y de calidad media respectivamente.

Para calibrar adecuadamente la ejecución de FRODOCK sobre el tipo de problemas presente en el conjunto de pruebas, 15 de sus casos (cinco por cada subconjunto) han sido empleados como conjunto de entrenamiento para determinar los parámetros óptimos del ajuste. Empleando este conjunto se ha determinado el valor de una serie de parámetros para mantener un balance óptimo entre precisión y eficiencia en la realización de los ajustes. De este modo, se ha establecido un intervalo traslacional de 2Å, mientras que el grado de la base de esféricos armónicos (*B*) que determina el intervalo de búsqueda rotacional es de 32 (siendo el intervalo rotacional de búsqueda asociado de menos de 6º). La representación armónica radial de los elementos a correlacionar se ha realizado mediante capas esféricas separadas radialmente 1Å y los tres términos energéticos (van der Waals, electroestática y desolvatación) han sido empleados con una relación de 1.0, 0.3 y 0.5 respectivamente.

Por último, los 76 casos han sido ajustados mediante estos parámetros 50 veces distintas partiendo de posicionamientos aleatorios del ligando con el fin de evitar pre-alineamientos previos entre las estructuras. Es importante destacar que la realización de estos ajustes es computacionalmente costosa. Dependiendo del tamaño de las proteínas involucradas, la aplicación FRODOCK puede requerir desde una hora para los casos más pequeños a tiempos cercanos a un día para los casos más grandes.

Fig. 5.7 muestra el acierto medio sobre el conjunto de pruebas empleado. Como puede observarse en el 90% de los casos se encuentra una solución aceptable ($RMSD_L$<10Å) entre las 10000 primeras soluciones. Este porcentaje se

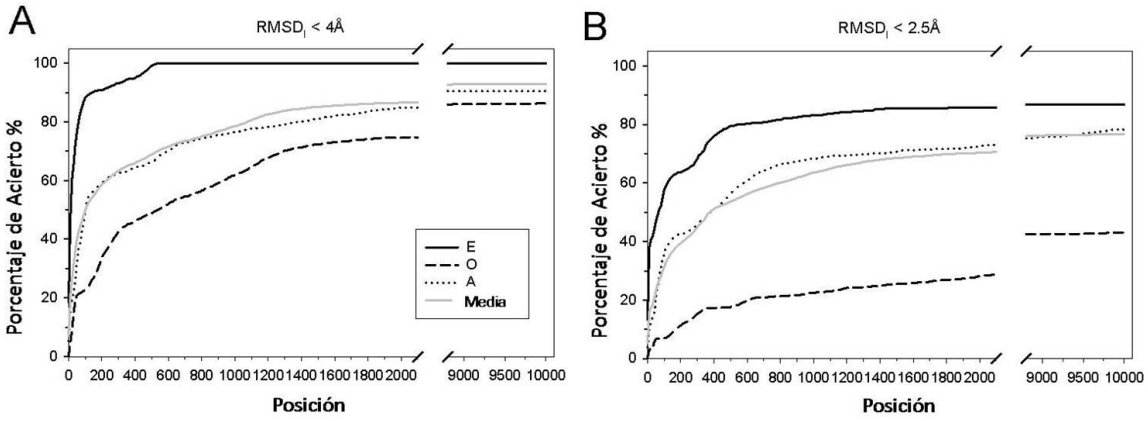

Fig.5.8 Porcentaje de acierto obtenido, evaluando mediante $RMSD_I$, en el conjunto de pruebas empleado para evaluar FRODOCK. Las gráficas **A)** y **B)** muestran los porcentajes para soluciones aceptables y de calidad media respectivamente.

reduce al 67% para soluciones de calidad media ($RMSD_L$<5Å). Si se limita el número de soluciones a las primeras 1000, entonces los porcentajes obtenidos son respectivamente de 78% y 53%. Cuando el límite es de 100 soluciones, los porcentajes son del 51% y del 30%. Si las soluciones son evaluadas empleando como criterio el $RMSD_I$ los porcentajes obtenidos son similares (ver **Fig. 5.8**). Por tanto, en más de la mitad de los casos es posible encontrar entre las 100 primeras soluciones una solución aceptable. Teniendo en cuenta la naturaleza del ajuste realizado, siendo éste un cribado inicial en el que cambios conformacionales no son considerados, estos resultados resultan muy positivos. Los porcentajes obtenidos son similares a los obtenidos con ZDOCK, considerada la aplicación estándar de ajuste rígido proteína-proteína de las actualmente disponibles [157]. Estudiando los resultados con más detalle se observa que para los distintos subconjuntos de pruebas la aplicación presenta comportamientos diferenciados. Así, para el subconjunto E, el comportamiento de FRODOCK es muy bueno ya que produce soluciones con $RMSD_L$ aceptable entre las 100 primeras para el 92% de los casos y soluciones de calidad media en el 57%. Estos porcentajes se reducen en el caso de A (50%,35%) y son mucho menores para el subconjunto O (22%,6%). Es bien conocido que la complementariedad de superficies es un criterio muy importante en interacciones enzima-substrato (grupo E) y anticuerpo-antígeno (grupo A). La contribución energética de más peso en FRODOCK es la de van der Waals que precisamente está relacionada con el grado de complementariedad. Por tanto, no es sorprendente que se obtenga una gran efectividad en estos dos casos. Sin embargo, en O, donde las interacciones entre proteínas tienen una naturaleza más heterogénea y compleja, este criterio tiene menos importancia. Por otro lado, en todos los casos se puede

	RMSD Ligando										RMSD Interfaz									
	< 10Å					< 5Å					<4Å					< 2.5Å				
	N	Pos	$RMSD_L$	f_{nat}	f_{not}	N	Pos	$RMSD_L$	f_{nat}	f_{not}	N	Pos	$RMSD_I$	f_{nat}	f_{not}	N	Pos	$RMSD_I$	f_{nat}	f_{not}
T11	16	26	9.81	0.14	0.59	-	-	-	-	-	6	241	3.01	0.37	0.49	-	-	-	-	-
T12	13	**1**	1.94	0.96	0.33	1	**1**	1.94	0.96	0.33	9	**1**	0.89	0.96	0.33	3	**1**	0.89	0.96	0.33
T13	70	23	7.90	0.40	0.57	5	68	3.91	0.72	0.21	87	23	2.66	0.40	0.57	18	68	0.94	0.72	0.21
T14	8	**1**	1.47	0.54	0.16	1	**1**	1.47	0.54	0.16	5	**1**	0.77	0.54	0.16	2	**1**	0.77	0.54	0.16
T18	6	261	7.36	0.70	1.10	-	-	-	-	-	8	261	2.62	0.70	1.10	1	3049	1.95	0.59	0.63
T18[a]	2	**10**	7.36	0.70	1.10	-	-	-	-	-	3	**10**	2.62	0.70	1.10	1	39	1.95	0.59	0.63
T19	13	**10**	6.86	0.44	0.48	1	401	4.89	0.73	0.19	16	**5**	3.45	0.46	0.79	1	401	1.24	0.73	0.19
T25	31	**3**	3.36	0.69	0.21	4	**3**	3.36	0.69	0.21	35	**3**	1.63	0.69	0.21	9	**3**	1.63	0.69	0.21
T26	11	224	3.92	0.29	0.33	2	224	3.92	0.29	0.33	9	224	2.08	0.29	0.33	3	224	2.08	0.29	0.33
T26[b]	7	32	3.92	0.29	0.33	2	32	3.92	0.29	0.33	7	32	2.08	0.29	0.33	3	32	2.08	0.29	0.33
T27	25	468	3.83	0.59	0.24	1	468	3.83	0.59	0.24	46	335	2.25	0.61	0.63	46	335	2.25	0.61	0.66
T27[c]	4	**2**	8.31	0.41	0.59	-	-	-	-	-	9	**2**	2.33	0.41	0.59	3	**3**	1.51	0.56	0.63

Tabla 5.3- Resultados obtenidos con FRODOCK en el ajuste de casos CAPRI. **N** indica el número de soluciones que cumplen el criterio de RMSD indicado. **Pos** es la posición de la primera solución que cumple el criterio de RMSD. Para el cálculo de los valores de $\mathbf{f_{nat}}$ y $\mathbf{f_{not}}$, se determino una interfaz según se describe en [156]. Los valores de RMSD han sido calculados empleando únicamente átomos Cα.
[a]Resultados obtenidos para T18 aplicando conocimiento previo acerca de la interacción. En este caso se conoce la implicación del residuo E70 de TAXI (receptor) con el residuo E179 de Niger Xilanasa (ligando). Las soluciones obtenidas son filtradas de forma que sólo se mantienen aquellas que presentan estos dos residuos a una distancia de 5Å.
[b] Resultados obtenidos para T26 aplicando conocimiento previo acerca de la interacción. En este caso se conoce la implicación de los residuos H246 y T292 de TolB (receptor) en la interfaz. Las soluciones obtenidas son filtradas de forma que sólo se mantienen aquellas que presentan estos dos residuos en la interfaz del complejo.
[c] Resultados obtenidos para T27 aplicando conocimiento previo acerca de la interacción. En este caso se conoce la implicación del residuo K14 de Hip2 (receptor) con el residuo C93 de Ubc9 (ligando). Las soluciones obtenidas son filtradas de forma que sólo se consideran soluciones con estos dos residuos a una distancia menor de 5Å.

encontrar una solución aceptable con la excepción de los casos *1BGX*, *1I4D*, *1SBB*, *1HE8* y *1IB1*, siendo estos casos conocidos por su difícil resolución (ver **Tabla A.1** y **Tabla A.2** en Apéndice). En otros casos, como *1A2K* o *1KLU*, FRODOCK encuentra una solución aceptable en posiciones muy altas, perdiéndose esta solución para alguno de los 50 posicionamientos aleatorias.

FRODOCK también ha sido validado empleando un conjunto de pruebas independientes extraído de experimentos CAPRI. Los experimentos CAPRI (*Critical Assessment of PRediction of Interactions*) son problemas de ajuste ciego que periódicamente son expuestos a la comunidad científica internacional con el fin de promover la competitividad en el avance de métodos de ajuste [158]. En **Tabla 5.3** se muestran los mejores posicionamientos encontrados en ajustes realizados con FRODOCK, pudiéndose observar que para la mayoría de los casos se obtienen soluciones aceptables por debajo de 100 posiciones. **Fig. 5.9** muestra el gran parecido que presentan los mejores ajustes obtenidos con FRODOCK y las soluciones conocidas.

Finalmente, para determinar la eficiencia de FRODOCK se realizó el ajuste del complejo *HyHel-5/lysozyme*. Este complejo ha sido previamente empleado como referencia para determinar la eficiencia de la aplicación Hex [159], siendo ésta la

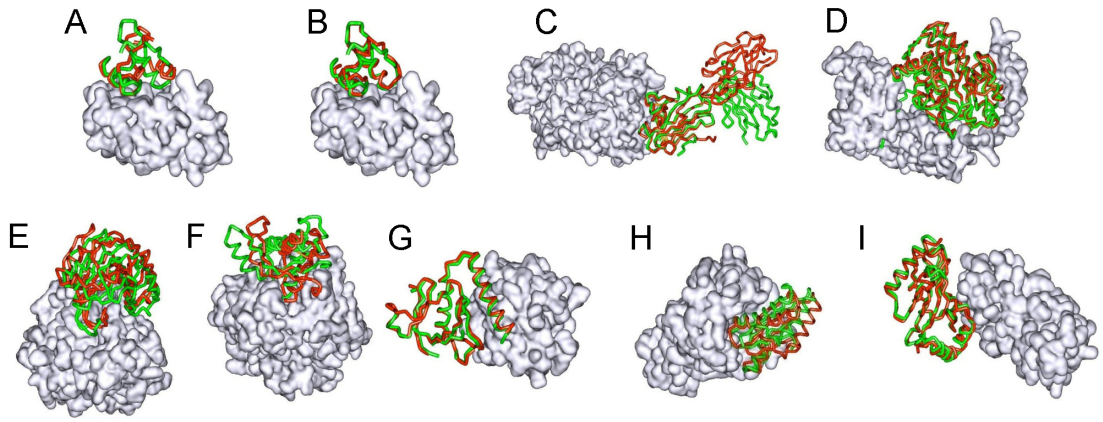

Fig. 5.9- Comparación entre los resultados obtenidos en los ejemplos de validación CAPRI mediante FRODOCK y las estructuras originales del complejo. En rojo se muestra la posición de la estructura del ligando en la mejor predicción obtenida con FRODOCK, en verde la posición de la estructura original. En gris se encuentra representado el mapa de superficie de la proteína receptor. Los objetivos CAPRI empleados son: T11(A); T12(B); T13(C); T14(D); T18(E); T19(F); T25(G); T26(H) y T27(I). Para una descripción detallada de estos casos véase http://www.ebi.ac.uk/msd-srv/capri/.

aplicación de ajuste proteína-proteína más rápida de las actualmente disponibles. En un computador Linux a 2.2MHz, Hex requiere un tiempo de cómputo de 27 minutos, incluyendo un paso posterior de refinamiento, para realizar el ajuste de este complejo, mientras que FRODOCK emplea un 10% más de tiempo. Sin embargo, si FRODOCK se emplea únicamente usando las interacciones energéticas de van der Walls y electroéstatica (que son las empleadas por Hex), el tiempo requerido se reduce a 18 minutos, obteniéndose una solución aceptable entre las 40 primeras soluciones, siendo éste un resultado comparable al obtenido con Hex. Si el tiempo de cómputo es comparado con ZDOCK, se observa una reducción de tiempo muy significativa, ya que esta aplicación requiere de más de dos horas. Además, la adaptación paralela de FRODOCK llevada a cabo (que puede ser empleada tanto en sistemas Grid como en *clusters*) permite reducir los tiempos de forma todavía más significativa.

Estos resultados confirman la validez de la aplicación FRODOCK, obteniéndose resultados como mínimo similares a los resultantes de la aplicación estándar de ajuste (ZDOCK). Además, FRODOCK permite obtener resultados en tiempos comparables a la aplicación más rápida existente (Hex). En conclusión, la combinación de efectividad y rapidez hacen de FRODOCK una aplicación altamente adecuada para la predicción de complejos proteína-proteína. Aún así, si bien FRODOCK muestra una alta eficiencia, para casos de ajuste en los que los tamaños de las proteínas son muy grandes, el tiempo necesario para resolver la búsqueda puede superar varias horas o incluso días. Por esta razón, la adaptación

paralela de la aplicación sistemas Grid resulta de gran utilidad, permitiendo obtener una alta productividad en el ajuste de proteínas.

6 Eficiencia del sistema cache

A pesar del alto grado de eficiencia mostrado por las aplicaciones bioinformáticas de ajuste desarrolladas, su aplicación en ciertos casos conlleva un alto coste computacional, no permitiendo obtener resultados en márgenes de tiempo adecuados al ser realizadas sobre computadores estándar. En este tipo de situaciones, los problemas de ajuste, como se ha indicado, pueden ser tratados como aplicaciones de alta productividad y adaptarse a entornos paralelos. Si bien alternativas como el uso de *clusters* o de computadoras multi-procesador son viables, la computación Grid resalta por la capacidad de proporcionar acceso puntual a grandes recursos computacionales que son difíciles de adquirir y mantener localmente. Esto es especialmente relevante en el caso de laboratorios o departamentos de biología estructural, donde este tipo de equipamiento no es habitual. Además, la realización de las adaptaciones paralelas de estas aplicaciones pueden verse favorecidas por la implantación del sistema cache desarrollado, dado que tanto los ejecutables de las aplicaciones como gran parte de los datos de entrada son comunes a todos las tareas paralelas.

En las siguientes secciones se muestran las mejoras en productividad que el sistema cache proporciona a la adaptación Grid de las aplicaciones bioinformáticas desarrolladas. Estas aplicaciones han sido empleadas sobre un entorno Grid donde el sistema de cache descrito en esta tesis doctoral ha sido implantado sobre un metaplanificador. De esta forma, tras una descripción de las características del entorno Grid, se realiza tanto el análisis de la eficiencia obtenida mediante el sistema cache sobre el proceso de transferencia de datos (empleando para ello la aplicación FRODOCK), como de la repercusión que esta mejora tiene en la ejecución global en condiciones experimentales reales (empleando tanto FRODOCK como ADP_EM). Por último, también se ha realizado un análisis de la

productividad empleando un *benchmark* de evaluación genérica de Grids. Con el fin de realizar este análisis, en una primera sección se introducen métricas y modelos de representación que permiten comparar el comportamiento del entorno Grid con y sin cache.

6.1 Caracterización de eficiencia

En esta sección se describen las métricas empleadas para comparar el comportamiento, con y sin funcionalidad cache, del sistema Grid en la ejecución de las aplicaciones bioinformáticas paralelas. Tal y como se indica en [160], para aplicaciones donde se ejecuta un conjunto de tareas paralelas empleando el mismo ejecutable sobre diferente información de entrada, el sistema Grid puede ser visto, desde el punto de vista computacional, como un conjunto de procesadores heterogéneos. Por tanto, el comportamiento del sistema para una aplicación, definido como el número de tareas finalizadas por unidad de tiempo, viene dado por:

$$n(t) = \sum_{i \in G} N_i \left\lfloor \frac{t}{T_i} \right\rfloor \tag{6.1}$$

Donde N_i es el número de nodos de cómputo del Grid que pueden realizar una tarea en T_i segundos. Esta función es discontinua e involucra a un alto número de términos, por lo que no proporciona una caracterización obvia. Sin embargo, es posible obtener una caracterización idealizada del sistema Grid al ajustar una recta sobre el comportamiento de éste. Los parámetros que definen esta recta proporcionarán información acerca de las características del sistema. Esta aproximación fue inicialmente propuesta por [161] para caracterizar arquitecturas de arrays homogéneos. De este modo, la recta de ajuste se define como:

$$n(t) = r_\infty t - n_{1/2} \tag{6.2}$$

La pendiente de la recta r_∞ define el *rendimiento asintótico* y determina el máximo rendimiento que puede ser alcanzado por el sistema Grid. Por su parte, $n_{1/2}$ se define como *distancia de medio rendimiento* e indica el número de tareas que deben de realizarse para obtener la mitad del *rendimiento asintótico*. Mediante estos parámetros es posible describir de forma idealizada el sistema Grid como un conjunto de $2n_{1/2}$ recursos homogéneos con un tiempo de ejecución

por tarea de $2n_{1/2}/r_\infty$ cada uno. Así mismo, estos parámetros pueden ser empleados para calcular el rendimiento ($r(n)$) y grado de homogeneidad (v) del sistema. El rendimiento del sistema está definido como el número de tareas completadas por segundo para un número finito de tareas y viene dado por la expresión:

$$r(n) = n(t)/t = \frac{r_\infty}{1 + n_{1/2}/n} \tag{6.3}$$

Donde *n* es el número finito de tareas. Por otra parte, el grado de homogeneidad se obtiene a partir de la *distancia de medio rendimiento* y el número de procesadores accesibles en el sistema Grid:

$$v = \frac{2n_{1/2}}{N} \tag{6.4}$$

Donde *N* es el número total de procesadores accesibles concurrentemente en el sistema Grid. El grado de homogeneidad indica el grado de similitud entre el número de procesadores real (*N*) y el número de procesadores aparente ($2n_{1/2}$) en la representación idealizada. Debido a que procesadores más rápidos pueden ofrecer mejores prestaciones que aquellos que son más lentos, el número de procesadores aparente será típicamente menor que el real. El grado de homogeneidad varía en el intervalo (0,1], siendo 1 en el caso idealizado en el que todos los procesadores del sistema se comporten del mismo modo (en cuyo caso el número de procesadores real y aparente será igual) y muy cercano a 0 cuando el comportamiento de los diferentes procesadores muestre una gran variación.

Estos parámetros proporcionan una caracterización idealizada del comportamiento de sistemas Grid heterogéneos, permitiendo su comparación. Así, un sistema que proporcione un *rendimiento asintótico* mayor será más productivo y un grado de homogeneidad alto indicará que un sistema mantiene un balance de la carga computacional sobre los diferentes procesadores más equilibrada que un sistema con un grado menor.

Por otro lado, si bien la caracterización descrita permite la comparación de diferentes sistemas Grid, ésta se realiza sobre el comportamiento general de los sistemas. Teniendo en cuenta que el establecimiento de la funcionalidad cache (única variante que va a ser introducida en el sistema Grid) afecta esencialmente al proceso de transferencia de ficheros de entrada, resulta adecuado establecer métricas para comparar la mejora obtenida en este proceso en cada uno de los

recursos computacionales del sistema. Así, si un recurso determinado realiza un número de tareas dado, el tiempo total empleado en transferir ficheros de entrada cuando no se emplea el sistema cache se establece por:

$$T_{nocache} = n_t(t_c + t_{nc}) \tag{6.5}$$

Donde n_t es el número de tareas realizadas en el recurso y t_c y t_{nc} son el tiempo de transferencia de los datos comunes a todas las tareas (y por tanto adecuados para ser almacenados en cache) y de los datos no comunes, respectivamente. Cuando el sistema cache es empleado, el tiempo de transferencia se calcula a partir de:

$$T_{cache} = n_t \cdot t_{nc} + n_f \cdot t_c \tag{6.6}$$

Siendo n_f el número de tareas cuya búsqueda en cache es infructuosa. Este valor se pondera con el tamaño de los ficheros, de forma que un fallo de acceso cache de un fichero tiene el doble de importancia que un fallo en el acceso a un fichero la mitad de grande. El valor de este tiempo estará limitado por el caso más favorable en el que los datos comunes sólo han de ser transferidos una única vez (n_f =1):

$$T_{mínimo} = n_t \cdot t_{nc} + t_c \tag{6.7}$$

Usando estas ecuaciones, la eficiencia en transferencia del sistema cache puede definirse como el grado de reducción obtenido en el tiempo de transferencia al emplear el sistema cache:

$$E_{cache} = 1 - \frac{n_t \cdot t_{nc} + n_f \cdot t_c}{n_t(t_{nc} + t_c)} \tag{6.8}$$

La eficiencia en transferencia será 0 cuando el sistema cache no permita reducir el tiempo de transferencia. Por otro lado, la máxima eficiencia esperable para un recurso se obtiene con $T_{mínimo}$:

$$E_{máximo} = 1 - \frac{n_t \cdot t_{nc} + t_c}{n_t(t_{nc} + t_c)} \tag{6.9}$$

6.2 Entorno Grid empleado

La infraestructura que ha sido empleada para realizar los experimentos está formada por recursos de la VO BIOMED dentro del proyecto EGEE, siendo GLite el middleware empleado en el sistema Grid. Con el fin de permitir una comparación de los resultados obtenidos entre los diferentes experimentos realizados se ha empleado siempre el mismo conjunto de recursos. Se han seleccionado nueve recursos disponibles en diferentes países europeos cuyas características están descritas en **Tabla 6.1**. El empleo de estos recursos en la organización virtual BIOMED presenta algunas limitaciones de uso orientadas a evitar saturación de las capacidades computacionales. Así, los recursos no aceptan más de diez tareas simultáneas por usuario. Debido a esto, la infraestructura Grid empleada está limitada a 90 procesadores concurrentes. Por otro lado, el metaplanificador GridWay ha sido configurado, siguiendo los parámetros establecidos por defecto, para realizar lanzamientos de 15 tareas concurrentes en períodos de 30 segundos. En cualquier caso, la variación de este esquema no afecta de forma significativa al comportamiento de las aplicaciones.

Entre las características de la infraestructura Grid empleada es importante resaltar que la mayoría de los recursos empleados no presentan un almacenamiento secundario común para todos sus nodos computacionales. Debido a esto, GridWay emplea un esquema de transferencia invertido en su comunicación con recursos EGEE. Este hecho obliga a que todos los experimentos realizados con apoyo de la funcionalidad cache hayan sido realizados empleando la política remota. Por ello y con el objetivo de estudiar el funcionamiento de la política centralizada también se han realizado pruebas empleando un recurso

Recursos	Dominio	SO	Arch.	Mhz	Nodos	Gestor local	Ubicación	Almacenamiento compartido
ce2	.egee.cesga.es	ScientificSLBer	i686	3000	112	lcgsge	España	No
Ce	.gina.sara.nl	ScientificSLBer	i686	2670	792	pbs	Holanda	Sí
clrlcgce03	.in2p3.fr	ScientificSLBer	i686	3200	240	lcgpbs	Francia	No
cox01	.grid.metu.edu.tr	ScientificSLBer	i686	1600	248	lcgpbs	Turquía	No
iut03auvergridce01	.univ.bpclermont.fr	ScientificSLBer	i686	3600	144	lcgpbs	Francia	No
paugrid1	.pamukkale.pl	ScientificSLBer	i686	1600	44	lcgpbs	Polonia	No
polgrid1	.in2p3.fr	ScientificSLBer	i686	2200	370	lcgpbs	Francia	No
trekker	.nikhef.nl	CentOSFinal	i686	2000	1150	pbs	Holanda	Sí
lcg38	.sinp.msu.ru	ScientificSLSL	i686	2700	104	lcgpbs	Rumanía	No
Aquila	local	Linux	i686	1995	4	pbs	Local	Sí

Tabla 6.1- Características de recursos Grid empleados.

local, también descrito en **Tabla 6.1**.

6.3 Estudio de eficiencia para FRODOCK

Con el fin de determinar la mejora que el sistema cache proporciona a la ejecución de la aplicación FRODOCK, se ha realizado un ajuste computacionalmente costoso empleando el entorno Grid indicado en la sección **6.2**. Los datos obtenidos a partir de esta ejecución han sido empleados para analizar el comportamiento del sistema Grid tanto en la transferencia de ficheros como en la ejecución general de la aplicación. Por último, también ha sido analizada la ejecución de un caso con carga computacional muy pequeña. De esta forma, se ha comprobado la adecuación del sistema cache para cualquier tamaño de este problema.

6.3.1 Descripción del problema empleado

Para evaluar la eficiencia del sistema cache sobre la aplicación FRODOCK se han realizado ejecuciones en el entorno Grid empleando uno de los casos más computacionalmente costosos del conjunto de pruebas empleado anteriormente para determinar la validez de la aplicación (véase sección **5.2**). En este caso, referenciado como *1N2C*, se ha de predecir a partir de las estructuras libres de las proteínas *Nitrogenasa Mo-Fe* y *Nitrogenasa Fe* la estructura del complejo que forman. Como se ha indicado en el capítulo anterior, la aplicación FRODOCK obtiene una solución aceptable para este caso entre sus cien primeras soluciones (ver **Fig. 6.1** y **Tabla A.1** en apéndice). La realización de este ajuste en un ordenador estándar (Pentium IV) requiere un tiempo de ejecución de alrededor de 17 horas.

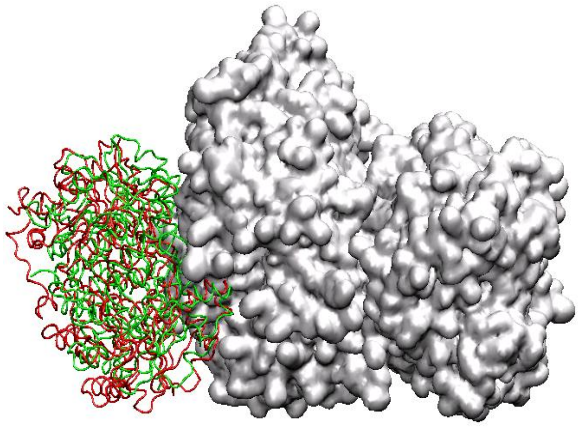

Fig. 6.1- Mejor ajuste encontrado empleando FRODOCK en el caso de prueba computacionalmente costoso *1NC2*. En rojo se muestra la primera predicción aceptable encontrada y en verde la estructura original. El mapa de superficie representa a la proteína receptor. Esta Predicción presenta una RMSD de 8Å con respecto a la posición correcta conocida. Esta orientación se encuentra entre las 100 primeras soluciones devueltas por FRODOCK.

La ejecución en el entorno Grid de cada una de las diferentes tareas paralelas necesarias para cubrir la búsqueda total en FRODOCK requiere de los siguientes ficheros de entrada:

- ***Ejecutable***: Todas las tareas realizan la misma computación sobre datos distintos. Por tanto, todas las tareas pueden emplear el mismo ejecutable (3MB) por lo que mantener este fichero en cache resulta beneficioso.
- ***Potenciales del receptor***: La descripción de la proteína establecida como receptor está contenida en diferentes mapas tridimensionales que representan sus diferentes potenciales de energía: van der Waals (11.5 MB), electroestático (11.5 MB) y desolvatación (7.4 MB). Además, en el caso de la desolvatación también es necesario una descripción del SASA de los átomos que lo componen (5.4MB). Todos estos ficheros son necesarios para todas las tareas y por tanto su introducción en cache es adecuada.
- ***Representación esférica del ligando***: Como ya se ha indicado en la sección **4.3.2** y se ilustra en **Fig. 4.20**, la información de la proteína que se establece como ligando puede ser pre-tratada. De esta forma, los términos dependientes del ligando en la integral (4.34) son procesados en un paso previo para obtener su representación radial armónica. Estos términos son, para cada correlación energética: la densidad electrónica para el término de van der Waals, la carga para el electroestático, el SASA para el de desolvatación del receptor y el potencial de desolvatación para el de desolvatación del ligando. Estas representaciones son empleadas por todas las tareas paralelas independientemente de las posiciones traslacionales que exploren. Así, la información acerca de dichas representaciones, convenientemente almacenada en dos ficheros (1.2MB), será introducida en cache.
- ***Posiciones traslacionales a explorar***: Cada tarea se encarga de explorar un conjunto de posiciones relativas del ligando respecto al receptor. De este modo, cada tarea tiene asignada la búsqueda de una parte diferente del espacio. Esta información, por tanto, no es introducida en cache al ser diferente para cada tarea. Por otro lado, el tamaño de estos ficheros varía en función del número de tareas paralelas (NT) en las que se divide la aplicación FRODOCK. Cuantas más sean las tareas en las que se divide la aplicación, menor número de posiciones corresponderán a cada una, siendo el tamaño de los ficheros más pequeño. Por el contrario, si hay un número pequeño de tareas paralelas, a éstas les corresponderán más posiciones y

los ficheros serán mayores. Así, el número de posiciones (NP) por tarea viene definido por

$$NP = \left\lceil \frac{NP_{total}}{NT} \right\rceil \tag{6.10}$$

Siendo NP_{total} el número de posiciones totales a explorar. Por ejemplo, en el caso *1N2C* donde hay 214188 puntos traslacionales a explorar, una división en 100 tareas conlleva que cada tarea explore aproximadamente 2142 (80K); mientras que si el número de tareas es de 350 cada tarea sólo explora 612 puntos (24K). Por tanto, el valor NT empleado resulta fundamental, ya que determinará la granularidad de la solución paralela adoptada. A valores NT pequeños les corresponderá una granularidad gruesa (NP será muy grande). Por el contrario, si NT es grande, la granularidad obtenida será muy fina (siendo NP muy pequeño).

Resulta interesante resaltar que ocho ficheros deben ser transferidos para cada una de las tareas paralelas cuando no se emplea el sistema cache. Sin embargo, cuando este sistema se usa, en el mejor de los casos, únicamente la colección de puntos a explorar debe transferirse, la cual presenta un tamaño muy pequeño con respecto al resto de ficheros.

6.3.2 Análisis de la eficiencia en transferencia

El uso del sistema cache afecta de forma directa a los procesos de transferencia involucrados en la ejecución de tareas. Por tanto, para determinar el impacto de la utilización del sistema cache sobre el tiempo de transferencia se han realizado pares de ejecuciones, empleando el sistema cache en una ejecución y desactivándolo en otra, de la aplicación FRODOCK sobre el problema *1N2C* variando diferentes aspectos. De este modo, se han realizado pares de ejecuciones empleando un solo recurso Grid y modificando el valor de NT (100, 200 y 300) con el fin de determinar la eficiencia obtenida con diferentes grados de granularidad. Estas pruebas han sido repetidas sobre recursos con diferentes arquitecturas de almacenamiento, siendo empleado el recurso *ce* como representante de la arquitectura de almacenamiento común a todos los nodos de cómputo y *paugrid1* como representante de la arquitectura de almacenamiento no común. Por último, al ser empleada la política cache remota sobre ambos recursos, por pertenecer al dominio de EGEE, se han repetido las pruebas sobre el

recurso local *aquila* para comprobar el funcionamiento de la política centralizada. En cada recurso, las pruebas han sido repetidas cinco veces con y sin cache.

A partir de las pruebas realizadas se han obtenido los datos necesarios para determinar las eficiencias en transferencia conseguidas en cada caso. En los casos en los que el sistema cache es empleado, t_{nc} se determina a partir del tiempo medio empleado por todas las tareas en transferir la información no común. t_c, por su parte, es obtenido como la suma de los tiempos medios necesarios para transferir cada fichero común cuando éste no se encuentra en cache. La suma de estos dos valores proporciona el tiempo de transferencia total para una tarea cuando ningún fichero se localiza en cache. A partir de la suma total del tiempo de transferencia empleado para todas las tareas ejecutadas, n_f puede ser determinado a partir de la ecuación (6.6). En lugar de comprobar directamente el número de fallos en los accesos a la cache, n_f se calcula obteniéndose, de esta forma, un valor ponderado de los datos no encontrados en cache. El número de fallos en el acceso a cache no tiene en cuenta el tamaño de los ficheros, de forma que un fallo en el acceso a un fichero pequeño tiene la misma importancia que el de un fichero grande. Calculando n_f mediante la ecuación (6.6) se proporciona un

Recursos	NT	Cache	T_c	T_{nc}	Ttransfer	Tiempo Total Trasf.	n_f	Tiempo Total Ejec.
Ce	100	Sí	17.30	0.30	17.6	48	**1.04**	44733
Ce	100	No	-	-	21.9	2196	-	45181
paugrid1	100	Sí	64.40	1.00	65.4	321	**3.43**	71529
paugrid1	100	No	-	-	65.1	6511	-	69212
aquila	100	Sí	29.80	2.56	31.6	376	**3.89**	58565
aquila	100	No	-	-	16.0	1603	**-**	58681
Ce	200	Sí	17.40	0,20	17.6	58	**1.04**	44611
Ce	200	No	-	-	22.3	4481	-	45171
paugrid1	200	Sí	64.80	0.80	65.6	479	**4.92**	70530
paugrid1	200	No	-	-	73.2	14654	-	70339
aquila	200	Sí	31.50	2.75	34.2	660	**3.49**	58810
aquila	200	No	-	-	16.6	3334	**-**	58798
Ce	300	Sí	17.20	0.30	17.5	109	**1.10**	45422
Ce	300	No	-	-	21.4	6462	-	45437
paugrid1	300	Sí	64.70	0.70	65.4	435	**3.48**	74024
paugrid1	300	No	-	-	69.7	20889	-	69871
aquila	300	Sí	31.50	3.25	34.7	1093	**3.74**	58885
aquila	300	No	-	-	1.5	4658	**-**	58928

Tabla 6.2- Datos experimentales obtenidos para el problema *1N2C* en diferentes recursos y con diferentes valores de NT. **Recurso** indica el recurso Grid usado en el experimento. **NT** corresponde al número de tareas paralelas en las que se ha dividido la ejecución. **Cache** indica si se ha empleado el sistema cache, **Tc** y **Tnc** corresponden a los tiempos medios de transferencia de los datos cacheables y no cacheables respectivamente (estos valores sólo se muestran para experimentos con cache). **Ttransfer** corresponde al tiempo medio de transferencia necesario para copiar todos los datos de entrada, este valor es igual a la suma de **Tc** y **Tnc** en los experimentos con cache. **Tiempo Total Trans.** corresponde al tiempo total empleado en operaciones de transferencia por todas las tareas. **n_f** corresponde al número medio de fallos de acceso a cache correctamente ponderado por el tamaño de los ficheros (únicamente para experimentos con cache). **Tiempo Total Ejec.** corresponde a la suma de los tiempos de cálculo de todas las tareas en el recurso.

valor más realista ya que el fallo en el acceso a un fichero estará ponderado por el tamaño de éste. Por último, en las pruebas en las que el sistema cache no se emplea, el tiempo total de transferencia para una tarea es calculado como la media de los tiempos necesarios para todas las tareas ejecutadas. En estos casos, los valores t_{nc}, t_c y n_f no son calculados. **Tabla 6.2** muestra los valores experimentales obtenidos para los diferentes recursos y valores de NT.

Tanto la eficiencia en transferencia obtenida como la máxima esperada han sido calculadas empleando estos valores sobre las ecuaciones (6.8) y (6.9). Los resultados obtenidos son mostrados en **Fig. 6.2**. Como puede observarse, los tres recursos muestran una eficiencia muy alta. En el caso de *ce* la máxima eficiencia esperada es alcanzada para los tres valores de NT. Este hecho se debe a la existencia de almacenamiento común en el recurso, ya que permite que todas las tareas tengan acceso a los ficheros comunes tras realizarse una sola operación de transferencia. Sin embargo, observando el valor de n_f, se puede apreciar que el número de transferencias de ficheros comunes excede levemente al de una única transferencia de ficheros. Efectivamente, para una sola transferencia de ficheros comunes el valor de n_f ha de ser 1. El hecho de que este valor sea ligeramente más alto indica que algunos de los ficheros son transferidos al menos por dos tareas diferentes. Esto es debido a la concurrencia en el lanzamiento de tareas. Al ser las tareas lanzadas por el planificador en grupos de 15, si las primeras comienzan su realización al mismo tiempo es posible que varias realicen la exploración del directorio cache antes de que sean instanciados algunos de los ficheros, de modo que todas ellas determinarán que deben transferir esos ficheros. En cualquier caso, esta concurrencia de transferencias es muy leve y,

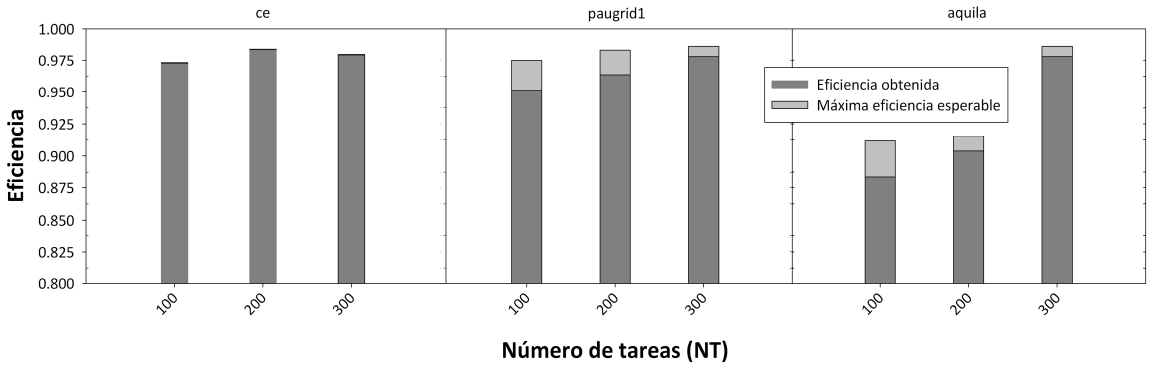

Fig.6.2- Eficiencia media (5 ejecuciones) obtenida sobre diferentes recursos en función del número de tareas (NT) en los que ha sido dividido el problema *1N2C*. Tanto en *ce* como en *paugrid1* ha sido empleada la política remota. En *aquila* ha sido empleada la centralizada.

como se ha indicado, permite alcanzar una eficiencia prácticamente idéntica a la máxima esperable.

Esta concurrencia en la transferencia también es responsable de que en el recurso *aquila* no se alcance la eficiencia esperada (aunque en este caso la concurrencia se produce en el acceso a la base de datos). En este caso, al ser el coste de transferir ficheros comunes t_c más alto y al ser el valor de n_f mayor, la eficiencia real se ve más afectada reduciéndose considerablemente con respecto a la prevista. n_f es en este caso mayor debido al uso de la política centralizada, la cual requiere la finalización de la fase de inicialización de tarea para modificar la base de datos, de forma que la ventana de tiempo en la que se puede producir concurrencia en la transferencia se ve ampliada.

En el caso del recurso *paugrid1*, tampoco se alcanza la máxima eficiencia debido a la falta de almacenamiento común. Sin embargo, a pesar de esto, la eficiencia obtenida resulta muy alta, encontrándose por encima de 0.95 para todos los valores de NT. Estos datos sugieren que, si bien no existe un almacenamiento común a todos los nodos de cómputo, sí están organizados en distintos subconjuntos donde el almacenamiento es compartido. Esto, junto con el hecho de que los mismos nodos de cómputo sean repetidamente seleccionados para realizar las tareas, justifica el bajo número de transferencias necesarias.

A partir de los resultados en los tres recursos, se evidencia la enorme mejora que supone, desde la perspectiva de las operaciones de transferencia, la presencia de almacenamiento compartido cuando se emplea un sistema cache.

En cuanto a la granularidad, es esperable que un aumento de NT, y por tanto una granularidad más fina, proporcione una eficiencia mayor. Al incrementarse el número de tareas, el tamaño del fichero que almacena las posiciones a explorar para cada tarea es menor. Al verse reducidos estos ficheros, que son la información no común a todas las tareas, el porcentaje de t_c en la transferencia total es aumentado mientras que el de t_{nc} es reducido. En consecuencia, la eliminación de transferencias de esta información común gracias al uso de cache supone una reducción mayor del tiempo de transferencia, y por tanto una mayor eficiencia. Este aumento de la eficiencia se observa en *paugrid1* y en *aquila*, sin embargo no se aprecia en el caso de *ce* (ver **Fig. 6.2**). Este último recurso presenta un valor para t_{nc} prácticamente despreciable para cualquier

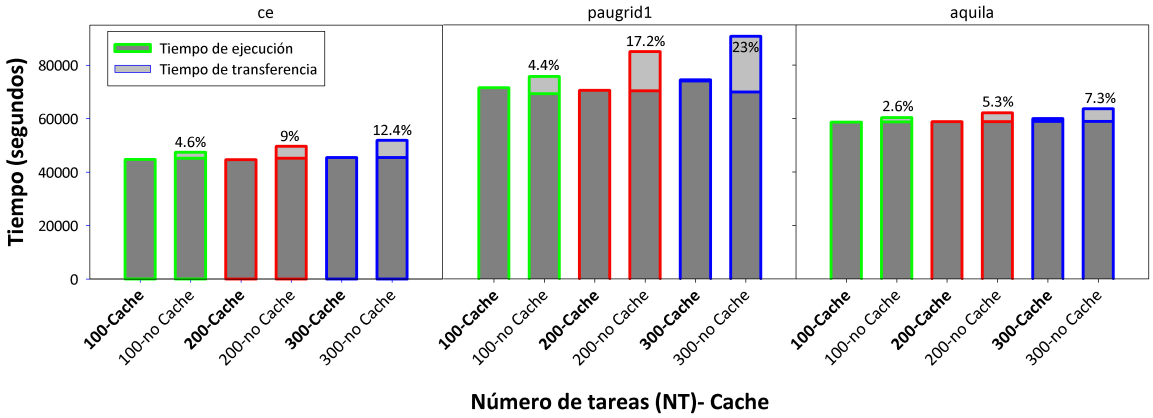

Fig.6.3- Tiempo operacional total (transferencia+ejecución) necesario para realizar todas las tareas en función del número de tareas (NT) en los que ha sido dividido el problema *1NC2.* Los porcentajes indican la proporción del tiempo de transferencia con respecto al tiempo de ejecución.

granularidad, de forma que su variación no afecta a la eficiencia conseguida. En los otros recursos, siendo t_{nc} mayor, su variación sí afecta a la eficiencia. Además, en estos recursos se hace evidente que la diferencia entre las eficiencias obtenidas y las máximas esperables se ve reducida al aumentar NT. Aunque el número de tareas se aumente, el valor de n_f se mantiene aproximadamente, de forma que el porcentaje de errores de acceso a la cache pierde importancia ante el creciente aumento de accesos a cache correctos. De esta forma, para un número de tareas infinito, la eficiencia obtenida terminaría alcanzando la máxima esperable.

El aumento de la granularidad conlleva un aumento de la transferencia que se evita en gran medida al emplearse el sistema cache. En **Fig. 6.3** se muestra el tiempo operacional total para todas las tareas en cada recurso calculado como la suma del tiempos de transferencia y ejecución totales sin tener en cuenta tiempos de espera en la ejecución de tareas. Como puede observarse, al no emplearse cache el tiempo de transferencia total se ve claramente aumentado al incrementar NT. Por el contrario, este tiempo es prácticamente despreciable en todos los casos cuando el sistema cache es empleado. Así, en *paugrid1* con NT=300 se obtiene, gracias al sistema cache, una reducción del tiempo operacional de alrededor del 20%. Esto es de gran relevancia, ya que permite determinar el grado de granularidad en la paralelización sin tener en cuenta el aumento de transferencia asociado a granularidades muy finas. La determinación de la granularidad óptima para una aplicación en un sistema Grid conlleva mantener un adecuado equilibrio entre saturación de los recursos disponibles, la concurrencia óptima en la

paralelización y el aumento de la transferencia. La eliminación de este último término facilitará su determinación.

Por último, a la vista de los resultados, se puede determinar que el comportamiento del sistema cache con las diferentes políticas es muy similar, dependiendo las variaciones en cada prueba a las diferentes características de los recursos empleados. La única diferencia notable recae en el valor n_f, pudiendo ser este ligeramente mayor cuando se emplea la política centralizada. Esto se debe a que la ventana de tiempo en la que se puede producir concurrencia en el acceso a la base de datos es mayor que la producida al hacer la comprobación directamente sobre los directorios cache. Sin embargo, si el número de tareas a ejecutar es muy alto, este incremento del valor de n_f será imperceptible.

6.3.3 Análisis de la mejora de productividad de la aplicación

Con el fin de determinar el comportamiento de la aplicación con funcionalidad cache sobre un entorno Grid compuesto por diferentes recursos (los nueve recursos de EGEE indicados), FRODOCK se ha empleado con y sin el sistema cache variando el número de divisiones de la aplicación en tareas paralelas. De este modo, se han empleado valores de NT iguales a 100, 150, 200, 250, 300 y 350 para determinar el comportamiento de la aplicación con distintos grados de granularidad. Para cada uno de estos casos el experimento ha sido repetido cinco veces con y sin cache. Además, las ejecuciones han sido realizadas de forma alterna, de modo que tras una ejecución con cache ha sido inmediatamente realizada una ejecución sin cache. Con ello, se ha minimizado en lo posible los efectos producidos por la variación en el comportamiento dinámico de los recursos. Como puede observarse en **Fig. 6.4**, en todos los casos salvo cuando NT=100 el tiempo de ejecución obtenido mediante el sistema cache es menor que el obtenido cuando el sistema cache no

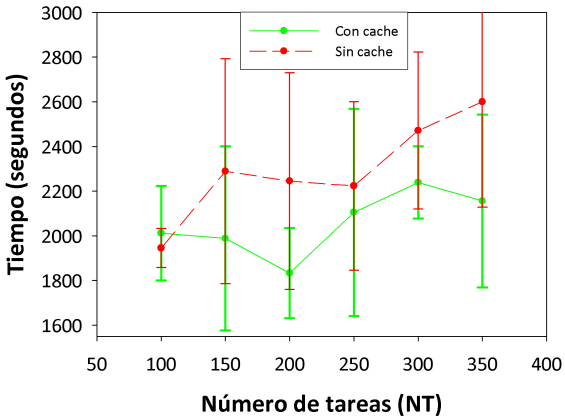

Fig.6.4- Media y desviación estándar de los tiempos de ejecución en función del número de tareas en los que el problema ha sido dividido (NT) para el caso *1N2C*. La línea discontinua en rojo corresponde a ejecuciones sobre un sistema Grid sin funcionalidad cache. La línea continua en verde corresponde a las ejecuciones sobre un sistema con cache.

es usado. En el caso de NT=100, el bajo número de tareas realizadas, muy cercano al número máximo de procesadores paralelos disponibles, hace que el rendimiento de la cache no sea muy elevado, impidiendo así obtener mejoras significativas. Entre los otros casos, el que presenta una mayor diferencia entre ejecución con cache y sin cache es el caso NT=200. Además, en este caso el tiempo de ejecución con cache es el menor de todos los obtenidos. Por tanto, es en este valor de NT donde se alcanza una granularidad para este problema mejor adaptada tanto a los recursos disponibles como a la presencia de la funcionalidad cache. Por ello, se ha tomado este caso como modelo para estudiar las mejoras proporcionadas por el sistema cache.

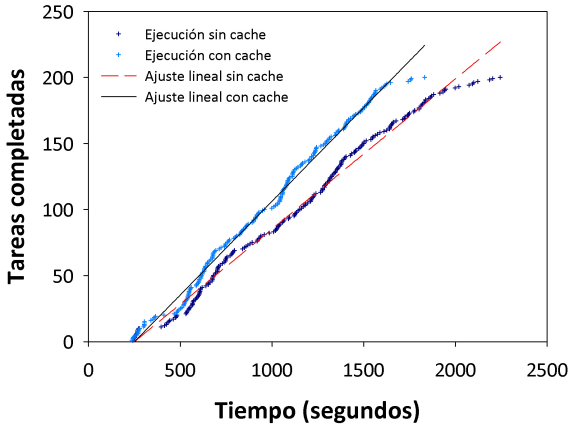

Fig. 6.5- Ajuste lineal del comportamiento (tareas realizadas por segundo) medio del sistema Grid para *1N2C* (NT=200) con y sin cache. Los puntos en azul claro muestran los datos experimentales obtenidos con cache sobre los que se ha realizado el ajuste (línea continua negra). Los puntos azul oscuro y la línea discontinua roja se corresponden respectivamente a los datos experimentales y ajuste obtenidos sin cache.

Para realizar este estudio, se han obtenido los valores de *rendimiento asintótico* y *distancia de medio rendimiento* para ambos casos mediante el ajuste lineal sobre el comportamiento (ecuación (6.2)) de ambos sistemas tal y como se observa en **Fig. 6.5**. Mediante este ajuste se obtienen los valores r_∞=0.14 y $n_{1/2}$=35.6 con el sistema cache mientras que, cuando no es empleado, los valores obtenidos son r_∞=0.11 y $n_{1/2}$=28.5. A partir de estos valores y empleando la ecuación (6.3) se ha calculado el rendimiento de ambos sistemas, tal y como puede observarse en **Fig. 6.6**, donde los rendimientos teóricos son comparados con los experimentales. Así, mientras que mediante el empleo de la cache se obtiene un rendimiento para 200 tareas de 0.119 tareas por segundo, sin cache el rendimiento decae a 0.096 tareas por segundo.

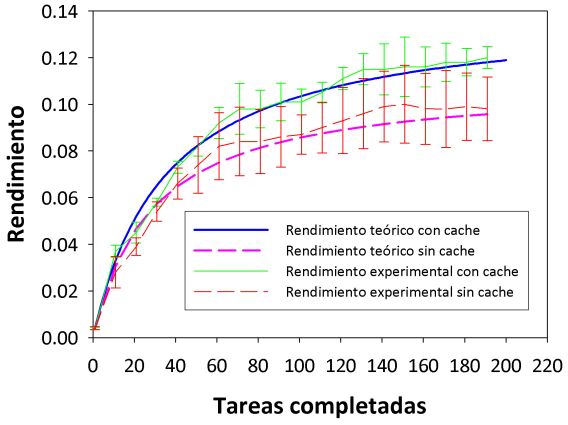

Fig. 6.6- Comparación entre rendimiento teórico y experimental medio para ejecuciones de FRODOCK con *1N2C* (NT=200) realizadas sobre el sistema Grid con y sin cache. Las barras de error indican desviación estándar en rendimiento experimental.

Estos valores muestran claramente la mejora que supone el empleo del sistema cache, ya que permite obtener hasta más del 22% de tareas ejecutadas en el mismo tiempo. Siguiendo el mismo procedimiento, se han calculado los rendimientos para todos los experimentos realizados con diferente NT. Los resultados exhibidos en **Tabla 6.3** muestran que, con la excepción del caso NT=100, siempre se obtienen mejores rendimientos cuando se emplea el sistema cache.

NT	Rendimiento con Cache	Rendimiento sin Cache	Ganancia %
100	0.0431	0.0439	-1.8
150	0.0658	0.0586	12.28
200	0.1192	0.0968	23.14
250	0.0933	0.0889	4.95
300	0.1229	0.1134	8.37
350	0.1492	0.1266	17.85

Tabla 6.3- Rendimientos obtenidos para diferente número de tareas (NT).

El ajuste al modelo lineal no sólo proporciona información sobre el rendimiento de los sistemas. Como se indicó anteriormente, la *distancia de medio rendimiento* proporciona información sobre el grado de homogeneidad del sistema Grid empleado. Empleando la ecuación (6.4), y teniendo en cuenta que en los experimentos realizados el valor de *N* efectivo es de 90 por las restricciones impuestas, el grado de homogeneidad que se obtiene para el caso en el que se emplea el sistema cache es de 0.79 mientras que cuando el sistema cache no es empleado se obtiene 0.63. Por tanto, los recursos del entorno Grid muestran un comportamiento más homogéneo con el sistema cache. Este aumento en la

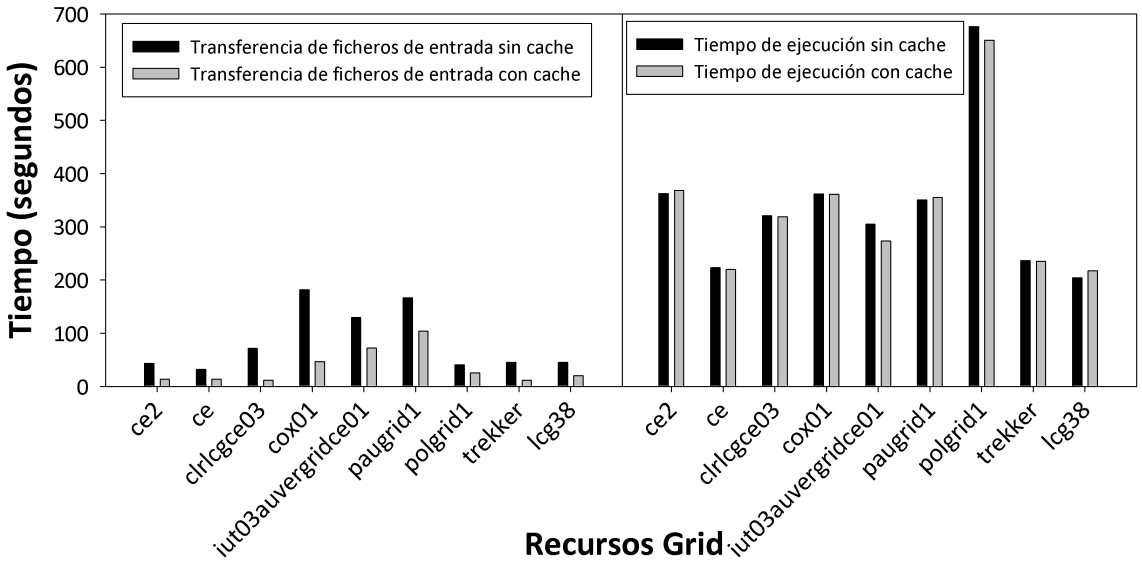

Fig. 6.7- Tiempos medios empleados, en cada recurso para transferir los ficheros de entrada (izquierda) y ejecutar (derecha) cada tarea de FRODOCK (caso *1N2C*, NT=200) sin (columnas en negro) y con (columnas en gris) cache.

homogeneidad se produce debido a la reducción en el tiempo de transferencia de los ficheros de entrada para todos los recursos. En **Fig. 6.7** puede observarse que, mientras los tiempos medios de ejecución son similares con y sin uso de cache, los tiempos de transferencia de ficheros de entrada se reducen gracias al sistema cache. Esta reducción conlleva un mayor parecido entre los tiempos de los diferentes recursos, siendo la desviación típica de éstos mucho menor con cache (32.7) que sin ella (59.0). Esta menor variación en el tiempo de transferencias repercute en el tiempo total, reduciendo las diferencias en éste para todos los recursos. En **Fig. 6.8** se muestran el número de tareas realizadas correctamente en cada recurso. Como puede apreciarse, la carga de tareas está ligeramente mejor repartida cuando se emplea el sistema cache (79.7 de desviación típica) que cuando no (83.0). Debido a que necesitan menos tiempo para realizar una tarea, recursos como *trekker* o *clrcgce03* que presentan una baja productividad, son más productivos al emplear la funcionalidad cache. Esto repercute sobre la carga de trabajo de los recursos más productivos (como *ce*) que se ve reducida al existir una menor cantidad de tareas que esperan ser asignadas a estos cursos.

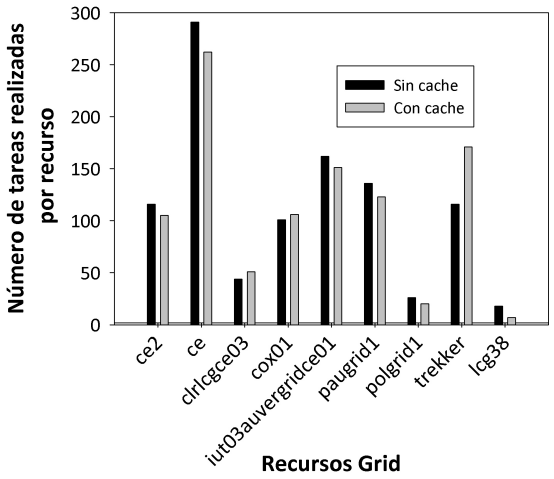

Fig. 6.8- Número de tareas realizadas en cada recurso entre los cinco experimentos con (columna gris) y sin (columna negra) cache (caso *1N2C,* NT=200).

En conclusión, el empleo del sistema cache no sólo permite un mayor rendimiento del entorno Grid, sino que además éste es mejor aprovechado, repartiéndose mejor las tareas entre los diferentes

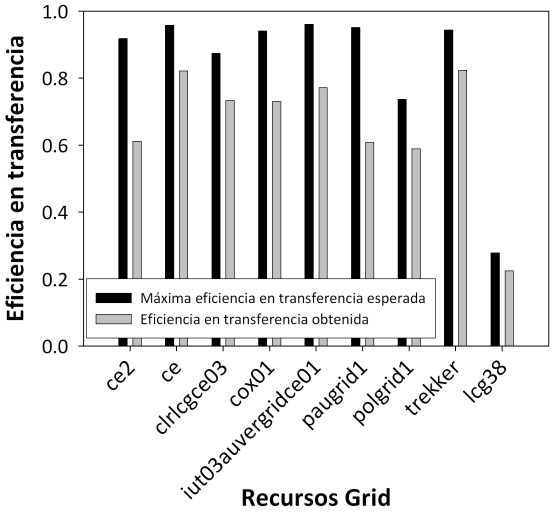

Fig. 6.9- Eficiencias en trasferencia obtenidas en los recursos (caso *1N2C, NT=200)*. Las columnas negras indican la máxima eficiencia que puede ser obtenida en cada recurso. Las columnas grises indican las eficiencias experimentales obtenidas.

recursos del entorno. De este modo, la aplicación no depende tanto de los recursos más productivos, distribuyéndose su realización de forma más homogénea en el conjunto de recursos.

Por último, en **Fig. 6.9** se muestran las eficiencias en transferencia obtenidas en todos los recursos empleados. En la gráfica son presentadas tanto la eficiencia obtenida como la máxima esperable. En todos los recursos, con la excepción de *lcg38*, se obtiene una eficiencia por encima del 50%. Esta medida indica reducciones de más de la mitad del tiempo de transferencia al emplear el sistema cache. Sin embargo, en todos los casos la eficiencia obtenida queda alejada de la máxima esperada. Como resulta lógico, aquellos recursos que presentan menor tasa de aciertos en cache (*ce2*, *paugrid1*, *polgrid1*, *lcg38*) también presentan una menor eficiencia; mientras que aquéllos que muestran un tasa de aciertos alta (*ce*, *trekken*) tienen eficiencias más altas. La tasa de aciertos (**Fig. 6.10**) para un recurso se determina como:

$$T_{acierto} = 1 - \frac{n_f}{n_{tareas}} \qquad (6.11)$$

Siendo n_{tareas} el número de tareas realizadas en el recurso. La diferencia en la tasa de acierto de los diferentes recursos se explica por el distinto número de tareas que cada uno realiza y a la variación de n_f. Sin embargo, debería haberse alcanzado la eficiencia máxima en los recursos donde el almacenamiento secundario es compartido (*ce*, *trekken*), ya que sólo la primera tarea debería de presentar fallos de acceso. La razón por la que la no se alcanza la eficiencia máxima viene dada por la configuración de ejecuciones del metaplanificador. Como se ha indicado anteriormente, éste realiza lanzamientos de 15 tareas simultáneas al entorno Grid. Si, inicialmente, varias tareas comienzan su ejecución al mismo tiempo sobre el mismo recurso, todas ellas apreciaran los mismos fallos de acceso a datos comunes. Por este motivo, existen

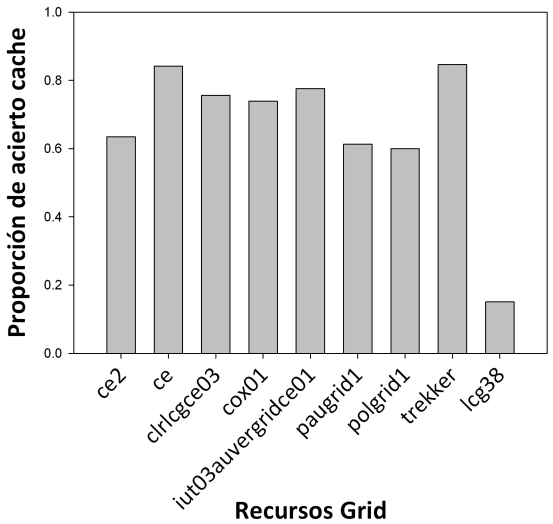

Fig. 6.10- Proporción de acierto en búsqueda de ficheros en cache para cada recurso (caso *1N2C, NT=200)*.

fallos en acceso a cache en más de una tarea, lo cual incrementa la tasa de fallos y disminuye la eficiencia. En cualquier caso, estos dos recursos (*ce*, *trekken*) son los que presentan mayor tasa de acierto, gracias a que realizan el mayor número de tareas, y por tanto una eficiencia más cercana a la máxima esperable.

En definitiva, el uso del sistema cache en la ejecución de la aplicación FRODOCK adaptada a un entorno Grid permite obtener un mayor rendimiento de los recursos. Además, proporciona un uso más equilibrado del sistema Grid al reducir la carga de trabajo sobre los recursos más productivos y aumentarla en los menos productivos. Esta mayor homogeneidad se produce gracias a la reducción en el tiempo de transferencia (a más de la mitad en la mayoría de recursos) proporcionada por el sistema cache.

6.3.4 Análisis en un problema de menor coste

En el apartado anterior se ha mostrado la mejora obtenida mediante el sistema cache sobre el caso computacionalmente más costoso del conjunto de validación empleado. Con el fin de establecer el incremento de la productividad también en problemas de coste computacional menor, se ha realizado el ajuste de uno de los casos más pequeños del conjunto de validación. Este caso, referenciado como *1PPE*, consiste en la determinación del complejo que conforman la proteína *Tripsina Bovina* y su inhibidor *CMTI-1*. Al ser ambas estructuras muy pequeñas, la resolución de este problema mediante FRODOCK es muy rápida, requiriendo menos de 40 minutos en un ordenador estándar. Este problema ha sido resuelto empleando los mismos nueve recursos Grid de EGEE utilizados anteriormente. Al ser un problema de menor dimensionalidad, tanto el tamaño de los ficheros como la carga computacional son menores. Esto repercute directamente en la granularidad más adecuada para realizar el ajuste en el sistema Grid. Al reducirse la computación, también se reducirá la carga computacional asociada a cada tarea paralela. Por

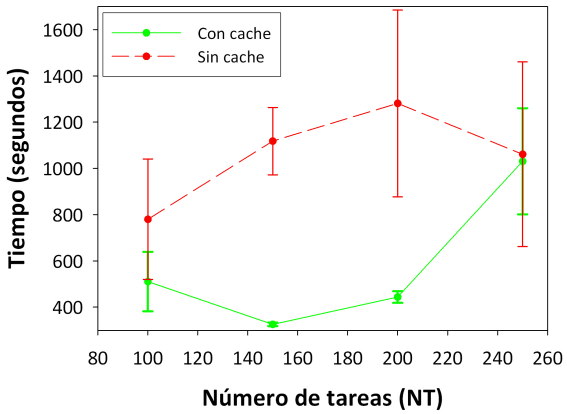

Fig. 6.11- Media y desviación estándar de los tiempos de ejecución en función del número de tareas en los que el problema ha sido dividido (NT) para el caso *1PPE*. La línea discontinua en rojo corresponde a ejecuciones sobre un sistema Grid sin funcionalidad cache. La línea continua en verde corresponde a las ejecuciones sobre un sistema con cache.

tanto, para obtener cargas computacionales similares al caso anterior con NT=200, será necesario emplear valores de NT menores, dividiendo la carga entre menos tareas paralelas. Por tanto, la granularidad más adecuada para este caso será más gruesa. Esto puede ser observado en la **Fig. 6.11**, donde se aprecia que con NT=150 se obtienen tiempos menores de ejecución completa. Por tanto, se empleará este valor de NT para realizar el análisis de la productividad. En este caso, el conjunto de siete ficheros comunes a todas las tareas tiene un tamaño de 6MB, mientras que los ficheros de posiciones de translación son sólo de 2.3KB. Por último, al igual que se hizo en el problema anterior, la ejecución ha sido repetida cinco veces empleando el sistema cache y cinco veces sin emplearlo.

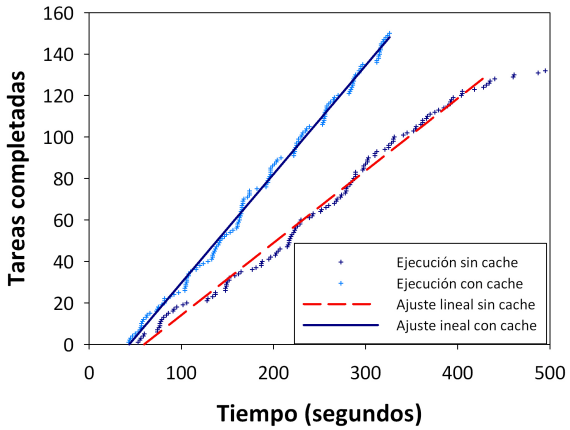

Fig. 6.12- Ajuste lineal del comportamiento (tareas realizadas por segundo) medio del sistema Grid para *1PPE* (NT=150) con y sin cache. Los puntos en azul claro muestran los datos experimentales obtenidos con cache sobre los que se ha realizado el ajuste (línea continua negra). Los puntos azul oscuro y la línea discontinua roja se corresponden respectivamente a los datos experimentales y ajuste obtenidos sin cache.

En **Fig. 6.12** y **Fig. 6.13** se muestran respectivamente los comportamientos y rendimientos del sistema Grid con y sin cache. En el caso de no emplearse la cache, la realización completa de todas las tareas requiere de un tiempo medio de 1118 segundos (19 minutos) frente a los 326 segundos (6 minutos) necesarios con cache. Sin embargo, se observa que el tiempo empleado en el caso sin cache está alterado por un excesivo retardo en las últimas tareas realizadas. Como puede observarse por el ajuste lineal realizado sobre el comportamiento experimental, si este retardo anómalo no se produjera, sería esperable un tiempo total cercano a los 600

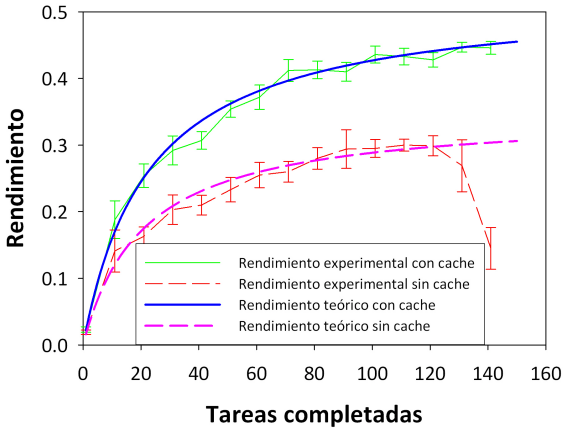

Fig. 6.13- Comparación entre rendimiento teórico y experimental medio para ejecuciones de FRODOCK con *1PPE* (NT=150) realizadas sobre el sistema Grid con y sin cache. Las barras de error indican desviación estándar en rendimiento experimental. Obsérvese la reducción del rendimiento experimental cuando no se emplea cache debido a la ejecución excesivamente retardada de las últimas tareas.

segundos (10 minutos). Por tanto, se puede determinar que el sistema cache proporciona una reducción teórica del tiempo de aproximadamente el 50%. Por otro lado, a partir de los ajustes lineales de los comportamientos se determina que el rendimiento del sistema sin cache está definido por r_∞=0.34 y $n_{1/2}$=20.8, mientras que cuando se emplea el sistema cache estos valores son r_∞=0.52 y $n_{1/2}$=22.8. Aplicando estos valores en la ecuación (6.3) el rendimiento ideal esperable para 150 tareas es de 0.30 tareas por segundo sin cache frente a 0.45 tareas por segundo con cache. Esto arroja un aumento en la productividad del 50% al emplear el sistema cache en consonancia con la reducción de tiempo ideal obtenida. Además el grado de homogeneidad vuelve a ser mayor al utilizar cache (0.304) que cuando no se emplea (0.27).

Por tanto, aunque para un caso de dimensiones tan reducidas el uso del sistema Grid no proporciona un incremento importante de la productividad, ya que su ejecución en un ordenador estándar requiere un tiempo de cómputo aceptable, la utilidad del sistema cache ha quedado demostrada también en casos de menor coste computacional al proporcionar prestaciones mayores a las obtenidas en el caso más computacionalmente costoso.

6.4 Estudio de eficiencia para ADP_EM

Como ya se ha indicado anteriormente, la adaptación de ADP_EM a computación Grid es especialmente interesante en casos donde esta aplicación se emplea como método de evaluación de modelos obtenidos por homología. En tales casos, el número de candidatos a ajustar (modelos obtenidos por homología) puede llegar fácilmente al orden de miles. El uso de un sistema Grid permite que la aplicación pueda ser ejecutada, para cada modelo, en recursos computacionales diferentes, realizándose concurrentemente la búsqueda de la estructura homóloga que mejor se aproxima a la real. Además, al realizarse los ajustes de todos los modelos homólogos sobre un mismo mapa tridimensional a baja resolución, gran parte de la información de entrada para todas las ejecuciones es común. Por ello, la implantación de la funcionalidad cache debe proporcionar una mejora significativa en la ejecución.

6.4.1 Descripción del problema empleado

Para la validación del sistema cache en este tipo de problemas, se ha empleado ADP_EM para determinar el mejor modelo, de entre 300 modelos de homología, de la estructura atómica conocida del complejo *E.Coli Malate Deshirogenasa* (entrada en *Protein Data Bank 2CMD*). Este test se encuentra incluido en el *benchmark* empleado en la sección **5.1.3**. El mapa ha sido filtrado a una resolución de 12Å y se le ha introducido ruido para reproducir condiciones experimentales. El ajuste de todos los modelos en un computador estándar requiere unas 19 horas para ser completado.

Cada ajuste de ADP_EM, realizado por una tarea independiente, requiere los siguientes ficheros de entrada:

- ***Ejecutable***: Compartido por todos las tareas (2.6 MB).
- ***Información sobre el mapa a baja resolución***: Al realizarse el ajuste sobre el mismo mapa tridimensional, todas las tareas pueden compartir esta información. Además la información sobre el mapa es pre-procesada para obtener la representación esférica de éste (3.1 MB).
- ***Posiciones de búsque*****da traslacional**: El conjunto inicial de posiciones donde puede ser colocadas las estructuras se determina *a priori* a partir del mapa a baja resolución (5.9 MB).
- ***Parámetros de búsqueda***: Diferentes parámetros (tales como dimensiones de la búsqueda, tamaño de muestreo traslacional, etc) que caracterizan la búsqueda del mejor ajuste son trasladados a las tareas a través de 3 ficheros (300 KB en total).
- ***Estructura atómica a ajustar***: A cada tarea se le transfiere un modelo atómico diferente para ser ajustado (Tamaño aproximado 179 KB).

Todos estos ficheros de entrada son susceptibles de ser mantenidos en cache al ser todos ellos compartidos por las tareas de ajuste, con la excepción de la estructura atómica a ajustar. Por tanto, la proporción de información de entrada exclusiva de cada tarea es my pequeña. Nótese que cada tarea realiza el ajuste completo de una estructura atómica y no se tiene la posibilidad de variar la carga computacional. Por tanto, la granularidad de la paralelización no puede ser modificada. Sin embargo, aunque no se ha considerado necesario, es posible aumentar el grado de granularidad dividiendo el ajuste de una estructura entre

varias tareas (o disminuirla reuniendo el ajuste de varias estructuras en una sola tarea).

6.4.2 Análisis de la mejora de productividad

La aplicación ha sido ejecutada sobre el entorno Grid anteriormente descrito (recursos de EGEE) con 300 modelos homólogos que han sido ajustados sobre un mapa simulado a partir de la estructura conocida del complejo. Como en el caso de la aplicación FRODOCK, con el fin de limitar el efecto de variaciones dinámicas en el comportamiento del entorno Grid, la aplicación ha sido ejecutada cinco veces con el sistema cache y cinco veces sin él. Estas ejecuciones se han intercalado para evitar en la medida de lo posible que las variaciones en el entorno afecten más a un tipo de ejecución. En **Fig. 6.14** se muestran el ajuste tanto de la estructura original como la del homólogo con mejor ajuste. Como puede así mismo observarse la comparación entre ambas estructuras muestra una gran similitud. Por otro lado, en la **Tabla 6.4** pueden observarse los tiempos empleados para cada una de las ejecuciones. Así, mediante el empleo del sistema cache se obtienen tiempos de ejecución que aproximadamente son hasta un 25% más cortos.

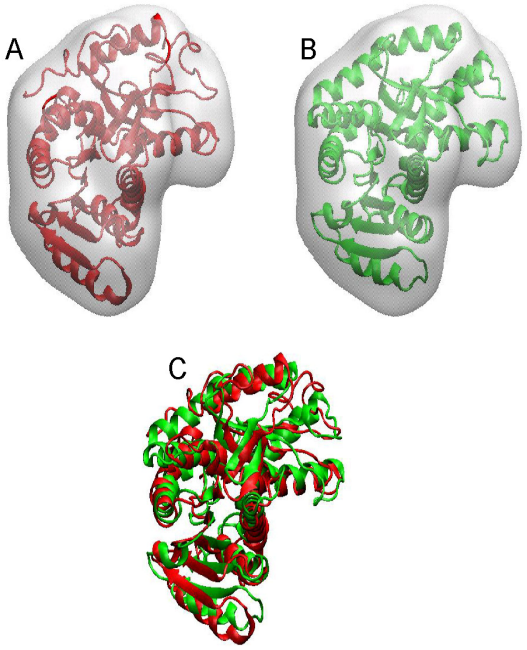

Fig. 6.14- Resultados de ajuste ADP_EM para evaluación de modelos de homología. **A)** Ajuste de estructura original de *2CMD* en mapa de baja resolución simulado. **B)** Ajuste del modelo con mayor correlación en el mapa de baja resolución. **C)** Comparación entre estructura original y mejor ajuste. Es reseñable la coincidencia espacial entre las hélices de las dos estructuras.

Si se establece el comportamiento (tareas finalizadas en función el tiempo transcurrido) a partir de los tiempos medios obtenidos para las cinco ejecuciones con cache, el ajuste realizado sobre éste arroja como valores representativos del sistema Grid un *rendimiento asintótico* r_∞ de 0.236 y una *distancia de medio rendimiento* $n_{1/2}$ de 46.86. Por el contrario, cuando no se emplea el sistema cache los parámetros que

Ejecuciones	1	2	3	4	5	media
Con Cache	1692	1481	1823	1471	1458	1585
Sin Cache	2678	2018	1844	2146	2270	2191

Tabla 6.4- Tiempos de ejecución para la aplicación ADP_EM en entorno Grid con y sin sistema cache.

definen el comportamiento idealizado del sistema son r_∞=0.199 y $n_{1/2}$=29.89 (ver **Fig. 6.15**). Empleando estos parámetros para determinar el rendimiento teórico de ambos sistemas se obtienen las curvas mostradas en **Fig. 6.16** obtenidas usando la ecuación (6.3). De esta forma, en el caso de 300 tareas se obtiene un rendimiento de 0.20 tareas por segundo con el sistema cache frente a 0.18 tareas por segundo sin él, lo que supone una mejora del 11%. Además, teniendo en cuenta que los rendimientos asintóticos determinan el máximo rendimiento que es posible obtener cuando el número de tareas tiende a infinito, se puede determinar que cuando el número de modelos creados por homología sea mayor (siendo esta la situación más habitual en condiciones experimentales reales) la mejora en el rendimiento alcanzará valores cercanos al 22%.

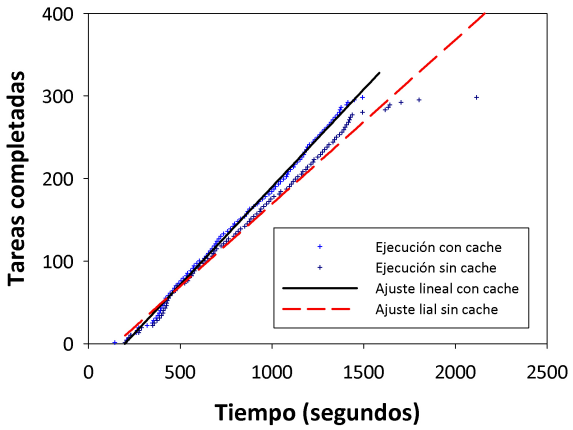

Fig. 6.15- Ajuste lineal del comportamiento (tareas realizadas por segundo) medio del sistema Grid para *2CMD* (NT=300) con y sin cache. Los puntos en azul claro muestran los datos experimentales obtenidos con cache sobre los que se ha realizado el ajuste (línea continua negra). Los puntos azul oscuro y la línea discontinua roja se corresponden respectivamente a los datos experimentales y ajuste obtenidos sin cache.

La mejora en el rendimiento también viene acompañada de un comportamiento más homogéneo del sistema, ya que a partir de la ecuación (6.4) se obtienen grados de homogeneidad de 1.041 y 0.664 respectivamente con y sin cache. Estos valores están afectados por pequeños desajustes entre los datos experimentales y el modelo lineal, no siendo posible un grado de homogeneidad por encima de 1. Como puede observarse en **Fig. 6.17** la reducción del tiempo medio de transferencia en todos los recursos del sistema Grid es significativa, lo cual repercute en un mayor parecido del tiempo total empleado por las tareas en todos los recursos. Esta

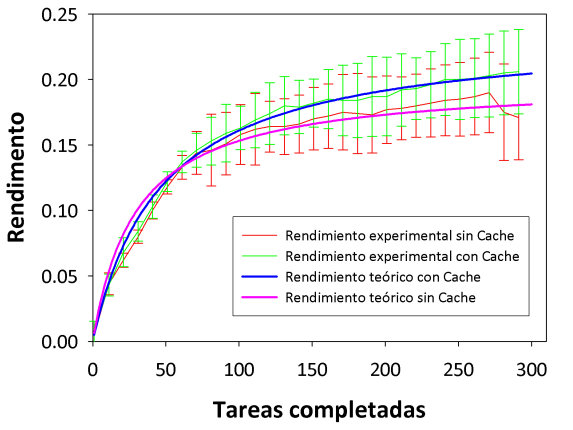

Fig. 6.16- Comparación entre rendimiento teórico y experimental medio para ejecuciones de FRODOCK con *2CMD* (NT=300) realizadas sobre el sistema Grid con y sin cache. Las barras de error indican desviación estándar en rendimiento experimental.

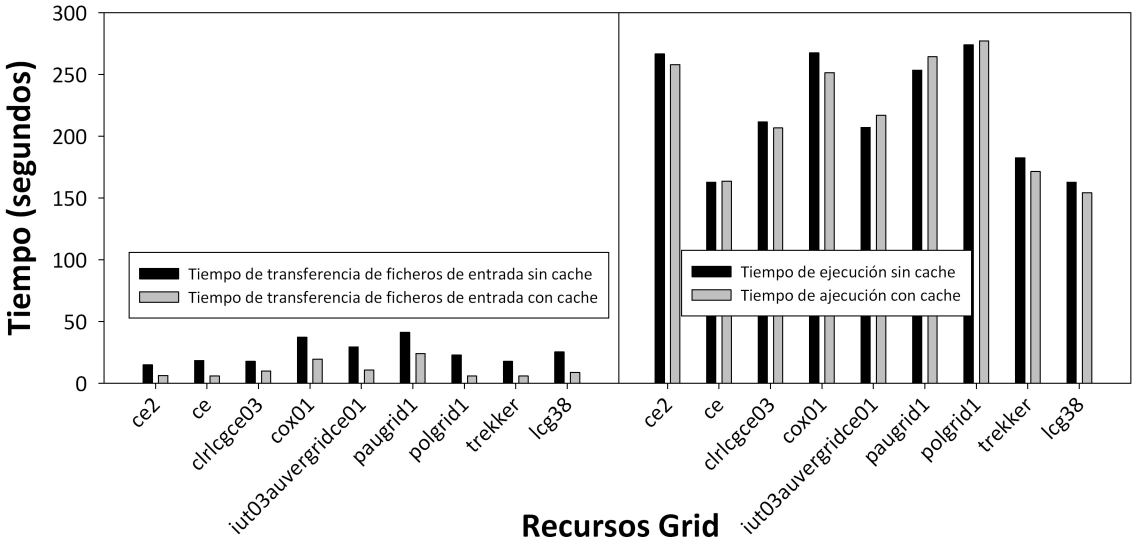

Fig. 6.17- Tiempos medios empleados, para cada recurso del sistema, en transferir los ficheros de entrada (izquierda) y ejecutar (derecha) cada tarea para ejecuciones de ADP_EM sin (columnas en negro) y con (columnas en gris) cache (*2CMD*. NT=300)

homogeneización de los recursos afecta al reparto de las tareas entre los recursos, que resulta ligeramente mejor compensada cuando se emplea el sistema cache, siendo la desviación típica de 52.83 con cache frente a 54.33 sin ella (véase **Fig. 6.18**). Por último, la mejora en el tiempo de transferencia de ficheros de entrada es consecuencia directa del éxito en la localización de ficheros en los directorios cache de los recursos (**Fig. 6.19**). Gracias a esto, se obtienen eficiencias de transferencia por encima del 50% en todos los recursos del sistema (**Fig. 6.20**).

Por tanto, al igual que en el caso de la aplicación FRODOCK, la aplicación de

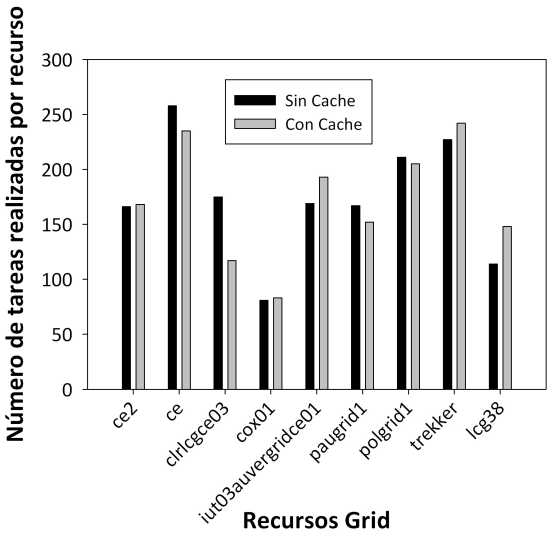

Fig. 6.18- Número de tareas ejecutadas en cada recurso entre los 5 experimentos ADP_EM con 300 modelos con (columnas grises) y sin (columnas negras) cache (2CMD).

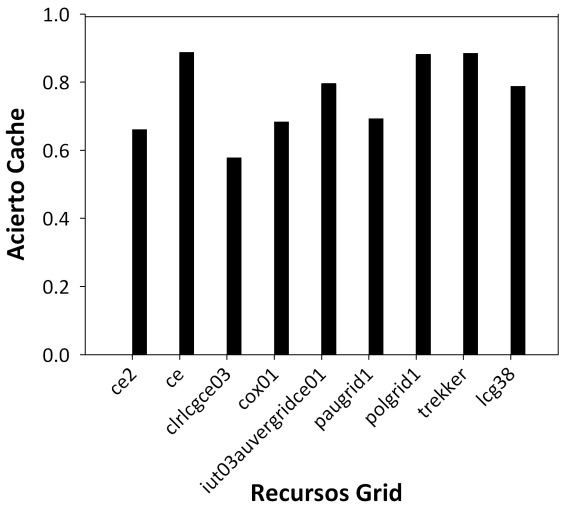

Fig. 6.19- Acierto medio en la búsqueda de datos (ficheros ponderados por tamaño) de entrada en cache (ADP_EM con 300 modelos. 2CMD).

este tipo de problemas en un entorno Grid puede verse sustancialmente beneficiado por el uso del sistema cache implementado, pudiéndose obtener hasta un 22% de ganancia en el tiempo de ejecución y haciéndose un uso del entorno Grid más equitativo entre todos los recursos.

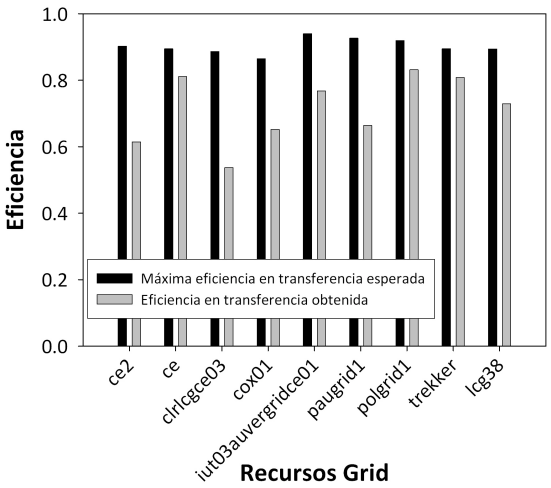

Fig. 6.20- Eficiencias en trasferencia obtenidas en los recursos (ADP_EM con 300 modelos. 2CMD). Las columnas negras indican la máxima eficiencia que puede ser obtenida en cada recurso. Las columnas grises indican las eficiencias experimentales obtenidas.

6.5 Estudio de eficiencia sobre benchmark Grid

Para finalizar el estudio de la mejora proporcionada por el sistema cache, se ha realizado un análisis más general que completa el realizado con las aplicaciones prácticas de ajuste bioinformático. La inexistencia de *benchmarks* que permitan el análisis de problemas de naturaleza tan específica como la de los tratados en esta tesis doctoral, ha llevado a la necesidad de readaptar *benchmarks* existentes para el análisis de eficiencia de sistemas Grids. Por tanto, se han realizado pruebas empleando un *benchmark* Grid NAS (NGB, del inglés *NAS Grid Benchmarks*) [162] convenientemente modificado para simular aplicaciones de alto rendimiento con alta reusabilidad de datos. Este tipo de pruebas están diseñados para generar cómputos muy extensos sin una transferencia de datos asociada, por lo que, aun modificados, su uso es poco adecuado para evaluar el comportamiento de un sistema cache. Sin embargo, su utilización permite determinar qué relación entre cómputo e información almacenada en cache permitirá a una aplicación mejorar su productividad gracias al sistema cache.

Los *benchmarcks* NAS definen grafos de flujo de datos que encapsulan instancias de códigos NPB (*NAS Parallel Benchmarks*) [163]. Estos NPB fueron inicialmente desarrollados para la evaluación del rendimiento de supercomputadores paralelos. Cada nodo en un grafo NGB representa una instancia NPB, definiendo los arcos comunicaciones entre los nodos. Un *benchmark* NGB se define por una serie de características:

- Los servicios que el sistema Grid debe proporcionar para realizar un modelo NGB son creación de tareas de cómputo y comunicación. NGB no asume

una implementación Grid particular, pudiéndose emplear distintas clases de *middleware*.

- Cada modelo NGB establece distintas familias de grafos de flujo de datos que caracterizan a distintos tipos de aplicaciones. De esta forma los modelos ED (*Embarrassingly Distributed*) permiten simular aplicaciones de alta productividad, los modelos HC (*Helical Chain*) representan cadenas de procesos repetitivos y dependientes unos de otros, los VP (*Visualization Pipe*) representan aplicaciones de visualización continua de procesamiento y los MB (*Mixed Bag*) representan aplicaciones de cómputo y visualización de naturaleza asimétrica.
- Existen distintas clases de modelos NGB. Cada clase determina un tamaño del problema diferente, es decir de las dimensiones de las matrices tridimensionales empleadas en los modelos. Si bien los códigos NPB no requieren ficheros de entrada, la entrada de datos puede ser simulada para determinar dependencias entre las distintas tareas.
- La granularidad de un *benchmark* puede ser controlada mediante el número de iteraciones que realizan los códigos NPB.

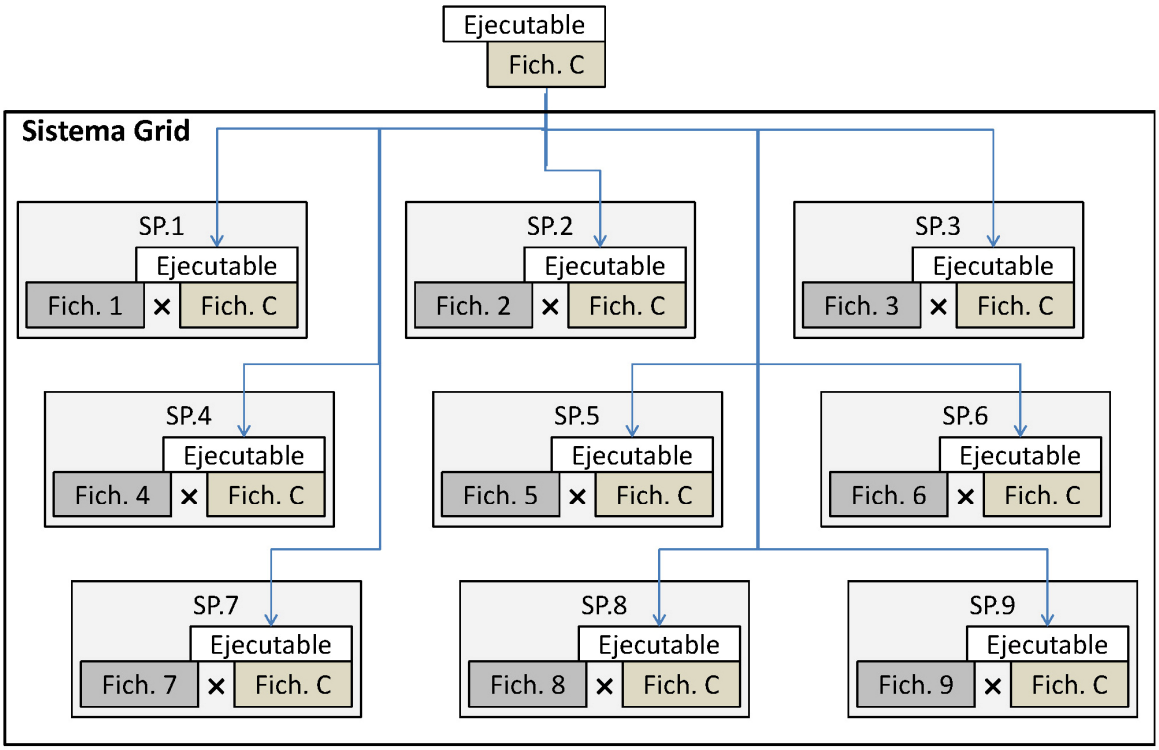

Fig. 6.21- Esquema de simulación de aplicación de alta productividad con reutilización de datos mediante problema ED. El ejecutable y un fichero de entrada (Fich. C) son comunes a todas las tareas, diferenciándose éstas por emplear un segundo fichero de entrada distinto. Los dos ficheros comunes pueden ser mantenidos en cache para mejorar el rendimiento del sistema.

Con el fin de simular una aplicación de alta productividad, ha sido empleado un modelo ED. Este modelo implica la realización de un conjunto de tareas NPB independientes, que ejecutan un código SP (*Scalar Pentadiagonal systems*) en la resolución de un sistemas de ecuaciones [164]. Este código realiza un cálculo sobre una matriz tridimensional de tamaño dependiente de la clase empleada y creada por el propio código. Como ya se ha puesto de manifiesto, el sistema cache resulta útil sobre aplicaciones de alta productividad con una alta reusabilidad de los datos de entrada. Para simular esta característica, se establece para cada tarea la necesidad de emplear dos ficheros de entrada. Cada uno de los ficheros contendrá la mitad de la matriz tridimensional, en flotantes de doble precisión, empleada en cada una de las clases. Mientras que uno de los ficheros será común a todas las tareas, el segundo será exclusivo para cada una de ellas. De esta forma se simulará la ejecución de una aplicación de alta productividad en donde una mitad de la información procesada es constante (primer fichero) mientras que la otra mitad varía (segundo fichero). De este modo, el mantenimiento en cache del segundo fichero, así como del ejecutable SP, proporcionará una mejora en la productividad al evitar repeticiones de transferencias. El esquema simplificado de la prueba diseñada puede verse en **Fig. 6.21**. Esta prueba ha sido realizada empleando las distintas clases (tamaño de problema) del *benchmark* NGB. La tabla **Tabla 6.5** muestra las dimensiones y tamaños de ficheros para las tres primeras clases. Por otro lado, el número de tareas empleado ha sido de 300 (suficiente para alcanzar la saturación el sistema Grid), realizando cada una de éstas una iteración del código ED. En cuanto al sistema Grid empleado, se han mantenido las mismas condiciones que en las secciones precedentes. Por último, tal y como se ha venido haciendo, se han

Clase	Dimensiones	Ficheros de entrada	Ejecutable
W	36x36x36	128KB x2	500KB
A	64x64x64	1024KB x2	500KB
B	102x102x102	4146KB x2	500KB

Tabla 6.5- Dimensiones de las matrices tridimensionales y tamaño de los ficheros para cada clase de problemas ED.

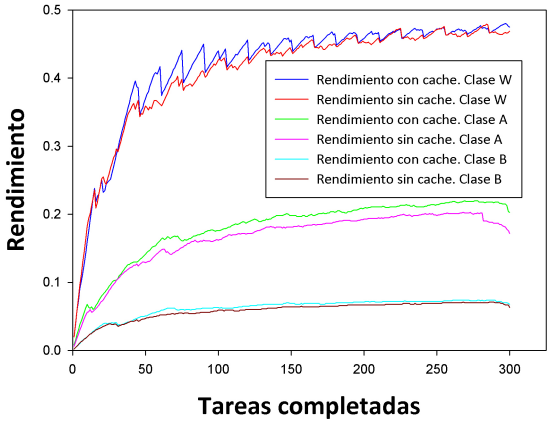

Fig. 6.22- Rendimientos medios obtenidos con y sin cache para las distintas clases del problema ED.

repetido las pruebas para las distintas clases cinco veces con y sin cache.

En la **Fig. 6.22** se muestran los rendimientos medios obtenidos para las clases W, A y B. Como resulta lógico, el rendimiento obtenido decrece conforme aumenta el tamaño del problema. Sin embargo, las diferencias producidas por el uso del sistema cache son únicamente relevantes en la clase intermedia A, donde r_∞ pasa de 0.2 sin cache a 0.22 con cache. Las otras clases presentan tamaños del problema que no permiten un óptimo aprovechamiento del sistema cache. En la clase W el tamaño excesivamente pequeño del fichero de entrada hace que el hecho de que las transferencias se reduzcan no tenga repercusión sobre el rendimiento. Por el contrario, en el caso B el incremento del tamaño de la información de entrada influye en mayor medida sobre el tiempo de cómputo, reduciendo tanto el rendimiento que la mejora obtenida por el uso de cache resulta insignificante. Para clases de mayor tamaño la reducción del rendimiento resulta más acentuada y la mejora obtenida por cache aún más imperceptible.

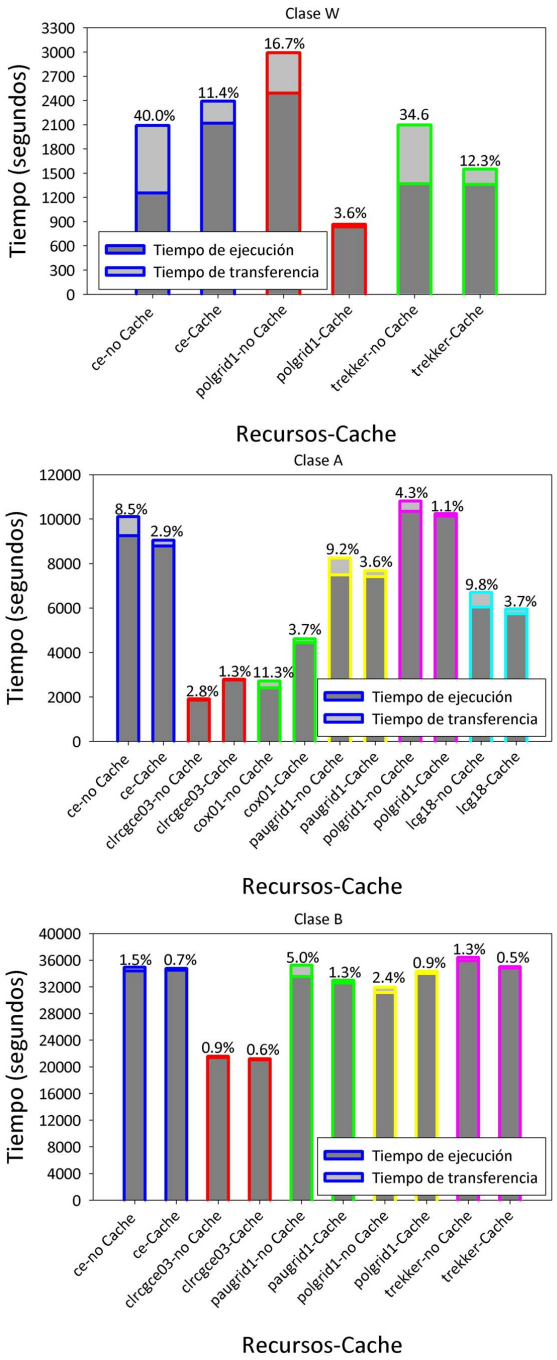

Fig. 6.23- Tiempo operacional total (transferencia + ejecución de todas las tareas realizadas) para cada clase del problema ED y en cada recurso con y sin cache (valores medios de las cinco ejecuciones). El valor sobre las columnas indica el porcentaje de tiempo empleado en realizar transferencias. Solo se muestran aquellos recursos que han sido realmente empleados en las ejecuciones.

Asimismo, en **Fig. 6.23** puede observarse, para cada clase, la relación entre el tiempo de ejecución de tareas y el tiempo de transferencia totales en cada uno de los recursos empleados en las pruebas realizadas. Si bien tanto los tiempos de ejecución como de transferencia varían de unos casos a otros debido a las

diferentes distribuciones de tareas entre los recursos, sí es posible comparar la relación existente entre ambos tiempos. De este modo, por ejemplo, se hace obvio que el porcentaje de tiempo empleado en transferencias decrece conforme aumenta el tamaño de la clase empleada. Esto es debido al mayor incremento del cómputo al aumentar el tamaño del problema. Por ello, en la clase B el porcentaje de tiempo empleado en transferencias es muy pequeño (menor del 5% en todos los recursos) incluso cuando no se emplea cache. La utilización del sistema cache, por tanto, no proporciona una mejora sustancial al reducir levemente este porcentaje. En la clase A la mejora es mucho más importante, ya que porcentajes alrededor del 9-11% en algunos recursos pasan a ser de alrededor del 3% cuando se emplea cache, lo que se refleja en la mejora de la productividad observada. En cuanto a la clase W, las reducciones del porcentaje de transferencia son mucho más visibles, pero los tiempos de transferencia son demasiado pequeños como para que su reducción afecte significativamente a la productividad.

Por tanto, la clase A proporciona la relación más adecuada entre tiempo de transferencia y tiempo de cómputo. Siendo el tamaño de los ficheros *cacheados* de aproximadamente 1.5 MB (1 de los ficheros de entrada y el ejecutable) y el tiempo de cálculo de una tarea requiere un tiempo medio de 140 segundos (si bien este valor varía en función del recurso empleado), se puede establecer una relación de aproximadamente 0.093 segundos de cálculo por *Kilo Byte cacheable* para obtener un mejora del rendimiento del 10% cuando los datos *cacheables* son un 60% de todos los datos a transmitir (mitad de la matriz y ejecutable). Una reducción de esta relación, reduciéndose el tiempo de computación o aumentándose los datos *cacheados*, supondrá un incremento de la mejora. Del mismo modo, un incremento del porcentaje de datos *cacheables* también debe repercutir en la mejora del rendimiento. Esta medida permite determinar la naturaleza de las aplicaciones de alta productividad que podrán beneficiarse del uso del sistema cache.

7 Conclusiones

7.1 Principales aportaciones

1) Se ha desarrollado un sistema cache para su utilización sobre tecnología Grid que presenta las siguientes características:

   - La implantación del sistema cache únicamente requiere modificaciones sobre un metaplanificador, no siendo necesario ningún tipo de alteración adicional en el sistema Grid. El diseño seguido permite su implantación sobre cualquier metaplanificador realizando pequeñas modificaciones en su funcionamiento. En concreto, se ha adaptado el metaplanificador GridWay sobre un entorno Grid (EGEE, VO Biomed).

   - El sistema permite mantener directorios en los recursos para almacenar ficheros *cacheables* siguiendo una filosofía "extremo a extremo". Con el fin de adaptar el sistema cache a diferentes esquemas de transferencia, dependientes del tipo de almacenamiento de los recursos, se han establecido dos políticas de manejo de los directorios cache, denominadas **Política centralizada** y **Política remota** respectivamente.

   - El sistema está enfocado a mejorar la productividad de aquellas aplicaciones de alta productividad que presenten un uso intensivo de la misma información de entrada.

2) Dos aplicaciones de ajuste bioinformático (ADP_EM para ajuste multi-resolución y FRODOCK para ajuste proteína-proteína) se han diseñado, validado y adaptado a la tecnología Grid:

   a. La aplicación ADP_EM ha sido creada para la realización del ajuste multi-resolución, realizando una paralelización en la que múltiples estructuras son ajustadas a un mismo mapa.

b. La aplicación FRODOCK ha sido desarrollada para realizar ajustes proteína-proteína con una paralelización del espacio de búsqueda traslacional.

c. Se ha comprobado que ambas aplicaciones son fiables y eficientes. La comparación del comportamiento de estas aplicaciones y el de aplicaciones similares existentes demuestra la importante mejora obtenida en la reducción de tiempo de cómputo.

3) La adaptación del sistema cache sobre el metaplanificador GridWay se ha empleado para estudiar su impacto sobre las aplicaciones de ajuste bioinformático. A partir de los experimentos realizados se ha podido constatar:

a. El aumento de productividad en ambas aplicaciones es evidente:

i. ***Ajuste multi-resolución***: Empleando ADP_EM en el ajuste de 300 modelos atómicos diferentes dentro de único mapa a baja resolución, el sistema cache permite mejoras en la productividad entre el 11-22%.

ii. ***Ajuste proteína-proteína***: La resolución de un problema de ajuste proteína-proteína empleando FRODOCK mejora su productividad entre el 22% y el 50% en función del tamaño del problema.

b. El comportamiento del entorno Grid se ve modificado en los siguientes aspectos:

i. En general, aquellos recursos con compartición de almacenamiento entre sus nodos computacionales o con tiempos de transferencia de datos muy altos presentan una mayor productividad al emplearse el sistema cache.

ii. La reducción en el tiempo de transferencia de datos obtenida por el sistema cache atenúa las diferencias entre recursos y produce un aumento en la homogeneidad del sistema. Esta mayor homogeneidad permite un reparto en la carga de trabajo entre recursos ligeramente mejor balanceado.

iii. La importante reducción de transferencias dada por el sistema cache permite el estudio de la paralelización de las aplicaciones sin tener en cuenta incrementos en la transferencia producidos por granularidades muy finas. Esto facilita la determinación de la granularidad óptima de aplicaciones en un sistema Grid.

c. El funcionamiento de ambas políticas de transferencia es similar. Sin embargo, la política centralizada proporciona al metaplanificador información acerca de la distribución de los datos *cacheados*. Un manejo adecuado de esta información puede permitir realizar una planificación más eficiente del uso de los recursos.

4) El sistema cache también ha sido analizado empleando un conjunto de pruebas estándar convenientemente adaptado. Este estudio permite determinar las proporciones entre tamaño de información compartida y tiempo de cómputo que deben de presentar las aplicaciones para obtener un aumento de la producción significativo mediante dicho sistema.

7.2 Publicaciones relacionadas

El diseño e implementación de las aplicaciones bioinformáticas de ajuste ha sido presentado, respectivamente, en las siguientes publicaciones:

- Garzon, J.I., Kovacs, J., Abagyan, R., and Chacon, P.(2007) *ADP_EM: fast exhaustive multi-resolution docking for high-throughput coverage.* Bioinformatics. **23**(4): p. 427-433.
- Garzon, J.I., Lopez-Blanco, J.R., Pons, C., Kovacs, J., Abagyan, R., Fernandez-Recio, J., and Chacon, P.(2009) *FRODOCK: a new approach for fast rotational protein-protein docking*. Bioinformatics. 25(19): p. 2544-2551.

La adaptación de la aplicación ADP_EM a la tecnología Grid fue inicialmente presentada en la comunicación:

- Garzon, J.I., Huedo, E., Montero, R.S., Llorente, I.M., and Chacon, P. *Grid Multi-Resolution Docking*. in *15th Euromicro Conference on Parallel, Distributed and Network-based Processing*. 2007. Naples, Italy.

Y Posteriormente publicada en:

- Garzon, J.I., Huedo, E., Montero, R.S., Llorente, I.M., and Chacon, P.(2007) *Adaptation of a Multi-Resolution Docking Bioinformatics Application to the Grid.* J. Softw. **2**(2): p. 1-10.

Por último, la adaptación de la aplicación FRODOCK, junto al diseño del sistema cache presentado en esta tesis doctoral y el análisis de su funcionalidad han sido tratados en:

- Garzon, J.I., Huedo, E., Montero, R.S., Llorente, I.M., and Chacon, P.(2009) *End-to-end Cache System for Grid Computing: Design and Efficiency Analysis of a High-throughput Bioinformatic Docking Application.* IJHPCA. **OnlineFirst**.

Además, durante el tiempo de realización de esta tesis doctoral también se ha estudiado, en colaboración con el laboratorio del Dr. Ruben Abagyan en The Script Research Institute (San Diego, California), la deformabilidad de estructuras atómicas mediante *Normal Mode Analysis*. Fruto de este estudio se presentó la publicación:

- Garzon, J.I., Kovacs, J., Abagyan, R., and Chacon, P.(2007) *DFprot: a webtool for predicting local chain deformability.* Bioinformatics. **23**(7): p. 901-902.

7.3 Trabajo futuro

Como se ha indicado, la política centralizada del sistema cache permite al metaplanificador almacenar información acerca de la localización de la información transmitida a los distintos recursos. Esta información puede ser de gran utilidad en la planificación de tareas, dándole prioridad a recursos que contengan la información necesaria. Sin embargo, introducir esta funcionalidad requerirá una modificación más profunda del metaplanificador, siendo necesario analizar el mecanismo del proceso de planificación. En este sentido, se pretende realizar un estudio futuro con el fin de modificar el sistema de planificación de GridWay empleando la información proporcionada por el sistema cache.

Asimismo, se realizará un estudio de posibles aplicaciones prácticas que puedan mejorar su rendimiento gracias al sistema cache.

Por último, el ajuste multi-resolución, convenientemente adaptado, puede ser empleado en el cribado virtual de drogas. Básicamente, el cribado virtual consiste en determinar el grado de ajuste de drogas en el sitio activo conocido de una proteína. De este modo, es posible discriminar drogas que puedan acoplarse a la proteína. El número de drogas que se testean en procesos de cribado virtual puede alcanzar el orden de millones, por tanto se hace necesario un proceso de ajuste eficiente además de una adecuada paralelización para emplear múltiples recursos Grid.

Apéndice

Tabla A.1- Resultados obtenidos con FRODOCK sobre el conjunto de pruebas estándar empleando como criterio de evaluación el cálculo de $RMSD_L$.

		$RMSD_L$<10Å						$RMSD_L$<5Å					
	T[a]	%[b]	N[c]	Rank[d]	$RMSD_L$[e]	f_{nat}[f]	f_{not}	%[b]	N[c]	Rank[d]	$RMSD_L$[e]	f_{nat}[f]	f_{not}
1AVX	ER	1.00	13.2	**2.0±1.0**	2.68±0.69	0.75±0.04	0.22	1.00	2.3	**2.0±1.0**	2.68±0.69	0.75±0.04	0.22
1AY7	ER	1.00	16.1	**10.5±1.3**	2.95±0.20	0.68±0.03	0.12	1.00	3.0	**10.5±1.3**	2.95±0.20	0.68±0.03	0.12
1BVN	ER	1.00	18.3	**1.7±0.6**	4.15±2.26	0.49±0.14	0.36	1.00	2.6	**2.4±1.0**	2.46±0.10	0.60±0.02	0.24
1CGI	ER	1.00	8.7	**17.7±2.2**	7.52±0.94	0.20±0.07	0.24	1.00	1.0	95.8±13.3	4.04±0.50	0.44±0.04	0.34
1D6R	ER	1.00	15.0	45.2±7.3	8.25±0.46	0.17±0.06	0.78	0.12	0.1	605.8±392.0	4.89±0.13	0.51±0.14	0.30
1DFJ	ER	1.00	20.1	**1.0±0.0**	6.21±0.41	0.66±0.02	1.11	1.00	3.5	**2.1±0.4**	4.45±0.57	0.46±0.07	0.86
1E6E	ER	1.00	9.9	**1.0±0.0**	8.24±0.08	0.24±0.02	0.58	0.92	1.6	109.6±52.8	3.83±0.84	0.38±0.08	0.28
1EAW	ER	1.00	23.3	**7.4±1.4**	3.82±0.61	0.74±0.02	0.40	1.00	2.9	**7.5±1.7**	3.73±0.13	0.74±0.02	0.39
1EWY	ER	1.00	11.6	**8.1±1.7**	8.52±0.32	0.20±0.02	0.92	1.00	2.8	30.6±4.4	4.40±0.54	0.31±0.09	0.68
1EZU	ER	1.00	11.6	116.6±71.3	7.41±1.47	0.43±0.03	0.30	0.16	0.2	1100±596.3	4.60±0.42	0.50±0.09	0.23
1F34	ER	1.00	10.1	**9.0±4.5**	6.47±0.85	0.32±0.09	0.31	0.94	1.2	380.0±671.8	4.17±0.48	0.29±0.08	0.16
1HIA	ER	1.00	11.8	**10.0±3.0**	8.86±0.16	0.14±0.02	1.16	-	-	-	-	-	-
1MAH	ER	1.00	16.6	**4.8±1.7**	2.49±0.27	0.77±0.06	0.47	1.00	3.7	**4.8±1.7**	2.49±0.27	0.77±0.06	0.47
1PPE	ER	1.00	21.5	**1.0±0.0**	0.95±0.07	0.86±0.01	0.27	1.00	1.7	**1.0±0.0**	0.95±0.07	0.86±0.01	0.27
1TMQ	ER	1.00	5.5	58.6±15.5	3.91±0.09	0.57±0.01	0.69	1.00	2.4	58.6±15.5	3.91±0.09	0.57±0.01	0.69
1UDI	ER	1.00	15.9	**10.9±1.9**	7.48±0.90	0.23±0.03	0.55	1.00	2.0	294.4±546.5	4.34±0.69	0.42±0.04	0.49
2MTA	ER	1.00	11.5	25.0±7.9	7.99±0.22	0.17±0.02	1.44	0.96	1.0	410.5±123.3	4.01±0.69	0.56±0.11	0.58
2PCC	ER	1.00	13.8	64.6±24.9	9.10±0.54	0.29±0.03	1.08	0.98	1.0	310.4±27.0	4.39±0.21	0.66±0.07	0.47
2SIC	ER	1.00	9.9	352±133	6.99±1.41	0.62±0.06	0.38	1.00	1.8	2019±1449	3.69±0.79	0.59±0.11	0.31
2SNI	ER	1.00	16.3	**4.4±1.1**	5.79±0.16	0.70±0.01	0.48	1.00	1.4	20.9±4.3	1.72±0.63	0.76±0.04	0.19
7CEI	ER	1.00	25.6	**1.6±0.6**	8.18±0.09	0.54±0.02	0.55	1.00	2.0	**17.5±4.8**	3.30±0.23	0.74±0.05	0.19
1ACB	EM	1.00	7.2	40.5±10.1	6.41±0.13	0.19±0.02	0.76	1.00	1.2	155.0±21.8	3.63±0.35	0.21±0.01	0.55
1KKL	EM	1.00	9.1	87.9±13.5	9.00±0.29	0.13±0.01	0.61	1.00	1.1	465.4±35.3	3.93±0.13	0.34±0.03	0.16
Avg	E	1.00	14.0	38.4±13.2	6.23±0.55	0.44±0.04	0.60	0.87	1.8	265.4±172.4	3.42±0.38	0.53±0.05	0.36
1AHW	AR	1.00	11.3	95.1±32.3	3.42±1.28	0.70±0.05	0.24	1.00	2.3	98.8±34.6	3.04±0.35	0.70±0.05	0.23
1BVK	AR	1.00	14.6	**12.1±15.6**	8.29±0.42	0.13±0.04	1.06	1.00	1.6	494.8±135.7	3.62±0.63	0.50±0.07	0.27
1DQJ	AR	1.00	1.9	1589±356	8.80±0.43	0.28±0.08	0.36	-	-	-	-	-	-
1E6J	AR	1.00	6.8	43.4±9.5	7.47±1.38	0.30±0.11	0.66	1.00	1.4	60.8±12.2	3.98±0.35	0.51±0.12	0.61
1JPS	AR	1.00	13.8	78.4±31.1	5.58±3.08	0.56±0.22	0.43	1.00	3.0	98.4±42.7	3.16±0.71	0.74±0.05	0.30
1MLC	AR	1.00	9.1	76.8±12.8	2.95±1.11	0.76±0.03	0.27	1.00	2.2	76.8±12.8	2.95±1.11	0.76±0.03	0.27
1VFB	AR	1.00	10.7	33.4±12.9	7.43±0.33	0.16±0.06	0.98	0.78	1.0	394.7±114.6	3.93±0.50	0.51±0.12	0.44
1WEJ	AM	1.00	4.0	45.6±6.6	4.20±0.74	0.63±0.03	0.47	1.00	2.0	46.8±8.4	3.63±0.81	0.66±0.07	0.44
2VIS	AM	-	-	-	-	-	-	-	-	-	-	-	-
1BJ1	AR	1.00	16.5	113.7±47.4	2.83±1.05	0.74±0.08	0.10	1.00	2.1	119.8±72.1	2.71±0.88	0.74±0.08	0.10
1FSK	AR	1.00	8.5	**2.8±1.1**	2.34±0.52	0.80±0.05	0.15	1.00	2.4	**2.8±1.1**	2.34±0.52	0.80±0.05	0.15
1I9R	AR	1.00	6.6	1453±452	6.46±2.24	0.58±0.07	0.29	0.92	1.0	2298±1391	3.41±0.80	0.60±0.05	0.29
1IQD	AR	1.00	17.4	199.7±47.2	7.21±1.41	0.41±0.11	0.55	1.00	2.1	656.5±494.2	3.86±1.27	0.68±0.07	0.22
1K4C	AR	1.00	9.2	3970±371	7.99±1.03	0.15±0.03	0.46	0.74	0.8	9183±893.2	4.28±0.42	0.42±0.03	0.17
1KXQ	AR	1.00	17.8	**1.0±0.0**	3.81±1.02	0.82±0.04	0.23	1.00	2.4	**2.7±2.7**	2.69±0.94	0.86±0.05	0.30
1NCA	AR	1.00	6.3	422.1±107.7	1.98±0.78	0.73±0.04	0.07	1.00	1.9	422.7±107.7	1.98±0.78	0.73±0.04	0.07
1NSN	AR	1.00	6.0	609.6±87.0	3.58±1.52	0.58±0.13	0.41	1.00	2.1	628.4±87.6	3.08±0.86	0.61±0.10	0.38
1QFW	AR	1.00	13.8	746.4±213.8	4.46±1.18	0.60±0.11	0.18	1.00	2.2	781.0±245.0	4.15±0.71	0.62±0.10	0.18
2QFW	AR	1.00	31.1	**7.4±2.9**	7.15±1.56	0.46±0.18	0.36	1.00	2.3	**20.0±9.9**	3.49±0.73	0.61±0.10	0.16
2JEL	AM	1.00	3.5	94.5±13.0	6.67±0.16	0.20±0.02	0.55	-	-	-	-	-	-
1BGX	AM	-	-	-	-	-	-	-	-	-	-	-	-
Avg	A	0.90	11.0	505.0±95.8	5.40±1.12	0.50±0.08	0.41	0.87	1.7	809.8±192.9	2.96±0.65	0.58±0.06	0.24
1A2K	OR	0.76	1.1	4539±995.3	9.64±0.31	0.54±0.15	0.68	-	-	-	-	-	-
1AK4	OR	0.12	0.1	8495±770.5	9.53±0.23	0.46±0.03	0.42	-	-	-	-	-	-
1AKJ	OR	1.00	11.8	38.9±14.4	8.90±0.14	0.20±0.01	1.88	0.70	0.9	28374±2120	4.18±0.82	0.55±0.10	0.63

1B6C	OR	1.00	29.0	**4.5±2.2**	3.89±0.58	0.76±0.02	0.51	1.00	2.8	**4.5±2.2**	3.89±0.58	0.76±0.02	0.51
1BUH	OR	1.00	11.7	590.4±271.3	7.77±1.43	0.17±0.08	0.41	0.76	1.1	2925±2899	3.67±0.72	0.40±0.13	0.23
1E96	OR	1.00	15.4	175.8±84.1	7.70±1.34	0.63±0.12	0.65	0.96	1.3	1342±1052	3.59±0.81	0.67±0.11	0.46
1F51	OR	1.00	9.1	29.7±8.4	8.00±0.61	0.34±0.05	0.61	1.00	2.9	151.0±35.6	3.89±0.39	0.44±0.12	0.30
1FC2	OR	0.98	2.9	1210±444.2	9.60±0.39	0.20±0.05	0.27	-	-	-	-	-	-
1FQJ	OR	1.00	12.2	222.0±42.7	6.71±0.84	0.43±0.05	0.39	-	-	-	-	-	-
1GCQ	OR	1.00	11.6	442.8±90.4	7.21±3.38	0.28±0.24	0.32	1.00	1.6	569.7±44.4	2.26±0.06	0.62±0.03	0.06
1GHQ	OR	-	-	-	-	-	-	-	-	-	-	-	-
1HE1	OR	1.00	9.0	1179±225.5	5.06±0.07	0.50±0.05	0.68	0.26	0.3	2354±2143	4.84±0.23	0.51±0.08	0.64
1I4D	OR	-	-	-	-	-	-	-	-	-	-	-	-
1KAC	OR	1.00	19.3	134.8±118.5	8.15±1.70	0.62±0.08	1.03	1.00	2.6	2074±435.8	3.92±0.53	0.54±0.04	0.37
1KLU	OR	0.76	0.9	8742±826.0	8.41±0.99	0.33±0.03	0.58	-	-	-	-	-	-
1KTZ	OR	1.00	15.0	305.2±45.5	3.89±0.21	0.77±0.03	0.33	1.00	2.8	305.2±45.5	3.89±0.21	0.77±0.03	0.33
1KXP	OR	1.00	15.3	114.3±59.9	6.29±1.74	0.35±0.10	0.28	0.98	1.4	200.4±91.1	4.54±0.34	0.30±0.08	0.25
1ML0	OR	1.00	18.9	34.9±4.3	5.45±3.20	0.49±0.26	0.44	1.00	2.9	37.3±5.9	3.19±0.77	0.66±0.09	0.33
1QA9	OR	1.00	2.2	4457±998.8	9.03±0.47	0.39±0.02	0.99	-	-	-	-	-	-
1RLB	OR	1.00	6.0	735.3±197.5	7.51±0.37	0.57±0.07	0.37	0.76	0.9	4228±1508	4.32±0.37	0.61±0.07	0.24
1SBB	OR	-	-	-	-	-	-	-	-	-	-	-	-
2BTF	OR	1.00	7.6	415.3±95.9	6.37±0.36	0.18±0.03	0.35	0.62	0.6	1684±435.3	3.88±0.44	0.31±0.02	0.23
1GP2	OM	1.00	3.8	463.2±645.8	9.10±0.66	0.12±0.01	0.20	0.42	0.5	4118±1264	4.73±0.41	0.12±0.01	0.15
1GRN	OM	1.00	12.5	131.3±39.6	6.37±0.53	0.39±0.02	0.47	0.78	1.0	4523±1103	4.11±0.65	0.34±0.07	0.29
1HE8	OM	0.22	0.2	6465±1828	9.17±0.48	0.17±0.07	1.34	-	-	-	-	-	-
1I2M	OM	1.00	8.5	77.4±22.7	8.51±0.66	0.19±0.03	0.73	0.16	0.2	4449±866.3	4.10±0.04	0.55±0.01	0.57
1IB1	OM	-	-	-	-	-	-	-	-	-	-	-	-
1IJK	OM	1.00	4.6	1196±146.0	5.91±0.59	0.48±0.03	0.61	0.34	0.4	4247±2793	3.67±0.92	0.38±0.12	0.45
1K5D	OM	1.00	9.0	368.5±69.2	7.20±0.72	0.15±0.02	0.54	0.36	0.4	5404±2289	4.21±0.38	0.13±0.02	0.11
1M10	OM	1.00	6.8	1005±450.6	8.61±1.10	0.19±0.06	0.71	0.40	0.4	3512±556.9	4.85±0.03	0.23±0.01	0.47
1N2C	OM	1.00	10.0	32.0±16.3	8.83±0.80	0.18±0.06	0.31	-	-	-	-	-	-
1WQ1	OM	1.00	1.4	202.5±53.0	5.24±1.10	0.38±0.03	0.32	0.28	0.3	210.9±54.2	3.48±0.33	0.42±0.03	0.31
Avg	O	0.81	9.1	1493±305.9	7.43±0.89	0.37±0.06	0.59	0.49	0.9	1615±705.3	2.83±0.32	0.33±0.04	0.25

[a]Clasificación de casos: E:Enzima-substrato; A:Anticuerpo-Antigeno; O: Otros, R:Cuerpo Rígido (dificultad baja), M:Dificultad media.
[b]Porcentaje de las 50 ejecuciones con al menos una solución encontrada en las 10000 primeras predicciones.
[c]Número medio de soluciones encontradas en las 50 ejecuciones con $RMSD_L$<10Å (izquierda) o $RMSD_L$<5Å (derecha) .
[d]Posición media de la primera solución encontrada con $RMSD_L$<10Å (izquierda) o $RMSD_L$<5Å (derecha). Los valores de límite RMSD corresponde al criterio CAPRI descrito en [156]. Casos sombreados en gris corresponden a soluciones encontradas entre las 100 primeras posiciones. Casos en negrita corresponden a soluciones encontradas entre las 20 primeras posiciones. El símbolo "-" indica que no se ha encontrado ninguna solución entre las 10000 primeras posiciones.
[e]El valor de $RMSD_L$ ha sido calculado empleando únicamente los átomos Cα de las estructuras.
[f]Los valores de la interfaz f_{nat} and f_{not} han sido calculados siguiendo el criterio CAPRI descrito en [156].

Tabla A.2- Resultados obtenidos con FRODOCK sobre el conjunto de pruebas estándar empleando como criterio de evaluación el cálculo de $RMSD_I$.

		RMSD$_I$<4Å						RMSD$_I$<2.5Å					
	T[a]	%[b]	N[c]	Rank[d]	RMSD$_I$[e]	f_{nat}[f]	f_{not}	%[b]	N[c]	Rank[d]	RMSD$_I$[e]	f_{nat}[f]	f_{not}
1AVX	ER	1.00	29.4	**2.0±1.0**	0.89±0.15	0.75±0.04	0.22	1.00	11.4	**2.0±1.0**	0.89±0.15	0.75±0.04	0.22
1AY7	ER	1.00	15.4	**10.5±1.3**	0.99±0.06	0.68±0.03	0.12	1.00	8.0	**10.5±1.3**	0.99±0.06	0.68±0.03	0.12
1BVN	ER	1.00	16.8	**1.7±0.6**	2.02±0.77	0.49±0.14	0.36	1.00	3.7	**2.3±1.0**	1.45±0.26	0.58±0.05	0.26
1CGI	ER	1.00	6.4	20.3±6.1	3.44±0.18	0.25±0.05	0.19	1.00	1.0	96.4±12.0	2.34±0.07	0.44±0.04	0.34
1D6R	ER	1.00	18.6	46.2±7.4	3.64±0.22	0.19±0.08	0.78	0.54	0.5	1072±1270	2.35±0.13	0.60±0.05	0.26
1DFJ	ER	1.00	15.4	**1.0±0.0**	2.43±0.10	0.66±0.02	1.11	1.00	3.5	**1.1±0.4**	2.42±0.10	0.65±0.04	1.10
1E6E	ER	1.00	8.8	**1.0±0.0**	2.88±0.05	0.24±0.02	0.58	1.00	1.8	97.0±32.4	1.93±0.27	0.38±0.07	0.26
1EAW	ER	1.00	28.1	**7.4±1.4**	1.26±0.16	0.74±0.02	0.40	1.00	6.6	**7.4±1.4**	1.26±0.16	0.74±0.02	0.40
1EWY	ER	1.00	6.7	29.5±4.0	2.57±0.24	0.28±0.06	0.71	1.00	2.4	37.3±7.4	1.82±0.37	0.56±0.16	0.54
1EZU	ER	1.00	16.4	83.8±35.5	2.88±0.66	0.38±0.08	0.35	1.00	2.3	273.3±252.9	2.10±0.20	0.44±0.05	0.24
1F34	ER	1.00	9.0	**9.0±4.5**	2.47±0.31	0.32±0.09	0.31	0.96	1.7	137.2±150.9	2.01±0.20	0.36±0.08	0.19
1HIA	ER	1.00	9.9	58.2±75.1	3.76±0.14	0.23±0.05	0.85	0.06	0.1	1724±252.6	2.47±0.01	0.34±0.01	0.66
1MAH	ER	1.00	15.0	**4.8±1.7**	1.02±0.11	0.77±0.06	0.47	1.00	5.0	**4.8±1.7**	1.02±0.11	0.77±0.06	0.47
1PPE	ER	1.00	24.9	**1.0±0.0**	0.57±0.02	0.86±0.01	0.27	1.00	7.0	**1.0±0.0**	0.57±0.02	0.86±0.01	0.27
1TMQ	ER	1.00	6.6	58.6±15.5	1.89±0.07	0.57±0.01	0.69	1.00	4.6	58.6±15.5	1.89±0.07	0.57±0.01	0.69
1UDI	ER	1.00	12.4	**12.8±6.4**	3.48±0.32	0.23±0.06	0.54	1.00	2.3	1507±1652	1.93±0.33	0.45±0.07	0.36
2MTA	ER	1.00	11.2	29.1±28.2	3.74±0.26	0.28±0.05	0.93	1.00	1.1	420.1±156.1	1.65±0.27	0.56±0.11	0.59
2PCC	ER	1.00	10.9	93.9±51.6	3.57±0.23	0.32±0.06	1.09	0.94	1.0	310.7±26.9	1.96±0.19	0.67±0.06	0.46
2SIC	ER	1.00	31.1	318.4±80.3	1.72±0.62	0.58±0.12	0.40	1.00	10.9	331.8±100.7	1.58±0.28	0.61±0.06	0.38
2SNI	ER	1.00	27.5	**4.4±1.1**	1.70±0.03	0.70±0.01	0.48	1.00	8.9	**4.4±1.1**	1.70±0.03	0.70±0.01	0.48
7CEI	ER	1.00	35.1	**1.6±0.6**	2.40±0.05	0.54±0.02	0.55	1.00	8.9	**1.8±1.1**	2.36±0.15	0.55±0.04	0.56
1ACB	EM	1.00	6.1	39.2±9.0	3.26±0.31	0.19±0.02	0.78	-	-	-	-	-	-
1KKL	EM	1.00	1.7	465.4±35.3	2.53±0.07	0.34±0.03	0.16	0.46	0.5	462.7±35.8	2.47±0.02	0.36±0.03	0.17
Avg	E	1.00	16.2	52.4±12.9	2.42±0.21	0.46±0.04	0.54	0.85	4.3	260.6±149.1	1.74±0.15	0.55±0.04	0.40
1AHW	AR	1.00	23.5	95.1±32.3	1.02±0.22	0.70±0.05	0.24	1.00	8.8	95.1±32.3	1.02±0.22	0.70±0.05	0.24
1BVK	AR	1.00	17.8	45.5±38.5	3.58±0.37	0.22±0.09	1.02	1.00	3.2	349.8±34.5	1.94±0.09	0.54±0.04	0.37
1DQJ	AR	1.00	4.1	1017±129.4	3.56±0.18	0.26±0.06	0.65	0.46	0.5	1835±464.0	2.05±0.08	0.42±0.02	0.30
1E6J	AR	1.00	9.3	45.6±10.8	3.24±0.81	0.37±0.16	0.64	1.00	3.6	57.6±8.4	2.10±0.15	0.54±0.12	0.63
1JPS	AR	1.00	37.7	35.0±15.5	3.41±0.66	0.26±0.12	0.98	1.00	14.7	92.8±37.2	0.95±0.27	0.71±0.09	0.31
1MLC	AR	1.00	16.6	76.8±12.8	1.16±0.08	0.76±0.03	0.27	1.00	7.7	76.8±12.8	1.16±0.08	0.76±0.03	0.27
1VFB	AR	1.00	11.9	80.2±23.6	3.59±0.16	0.28±0.03	0.87	1.00	2.4	401.2±130.8	2.02±0.26	0.56±0.06	0.43
1WEJ	AM	1.00	4.7	45.6±6.6	1.21±0.16	0.63±0.03	0.47	1.00	3.6	45.6±6.6	1.21±0.16	0.63±0.03	0.47
2VIS	AM	1.00	6.4	1369±460.0	3.42±0.20	0.22±0.03	1.35	0.66	1.1	6469±1716	2.03±0.23	0.44±0.06	0.78
1BJ1	AR	1.00	70.5	113.7±47.4	0.85±0.14	0.74±0.08	0.10	1.00	28.6	113.7±47.4	0.85±0.14	0.74±0.08	0.10
1FSK	AR	1.00	8.6	**2.8±1.1**	0.97±0.15	0.80±0.05	0.15	1.00	7.3	**2.8±1.1**	0.97±0.15	0.80±0.05	0.15
1I9R	AR	1.00	19.4	1103±314.6	2.57±0.64	0.40±0.16	0.38	1.00	5.9	1253.±402.3	2.08±0.21	0.54±0.08	0.29
1IQD	AR	1.00	44.8	145.8±25.7	3.39±0.28	0.38±0.04	0.61	1.00	8.0	288.2±64.6	1.51±0.61	0.64±0.10	0.38
1K4C	AR	1.00	19.0	3443±354.8	3.59±0.21	0.14±0.03	0.38	1.00	4.2	5656±483.6	2.04±0.21	0.34±0.03	0.14
1KXQ	AR	1.00	17.3	**1.0±0.0**	1.25±0.22	0.82±0.04	0.23	1.00	6.6	**1.0±0.0**	1.25±0.22	0.82±0.04	0.23
1NCA	AR	1.00	12.3	420.9±108.0	0.81±0.38	0.72±0.08	0.08	1.00	6.3	422.1±107.7	0.76±0.15	0.73±0.04	0.07
1NSN	AR	1.00	6.2	609.6±87.0	1.78±0.59	0.58±0.13	0.41	1.00	2.1	630.4±92.1	1.56±0.29	0.62±0.08	0.38
1QFW	AR	1.00	47.2	742.1±209.7	1.35±0.38	0.58±0.11	0.19	1.00	18.9	746.1±213.3	1.29±0.26	0.59±0.11	0.18
2QFW	AR	1.00	128.7	**6.6±3.0**	2.64±0.67	0.39±0.18	0.42	1.00	21.8	**9.6±5.7**	1.86±0.28	0.62±0.12	0.27
2JEL	AM	1.00	3.7	94.5±13.0	3.43±0.08	0.20±0.02	0.55	-	-	-	-	-	-
1BGX	AM	-	-	-	-	-	-	-	-	-	-	-	-
Avg	A	0.95	25.5	474.7±94.7	2.34±0.33	0.47±0.08	0.50	0.91	7.8	927.4±193.0	1.43±0.20	0.59±0.06	0.30

1A2K	OR	1.00	7.0	1204±172.6	3.50±0.10	0.26±0.02	0.23	0.32	0.3	4858±828.4	2.37±0.07	0.41±0.02	0.22
1AK4	OR	1.00	27.2	909.5±500.	3.78±0.15	0.29±0.10	0.76	0.90	1.5	5924±1647	2.28±0.13	0.50±0.05	0.44
1AKJ	OR	1.00	13.8	169.8±40.9	2.37±0.58	0.57±0.11	0.75	1.00	4.0	187.1±51.2	2.03±0.17	0.65±0.02	0.69
1B6C	OR	1.00	18.8	**4.5±2.2**	2.38±0.15	0.76±0.02	0.51	0.80	0.9	**14.0±31.8**	2.31±0.13	0.76±0.04	0.48
1BUH	OR	1.00	9.3	829.6±220.	2.54±0.59	0.25±0.08	0.30	1.00	2.4	937.1±95.3	2.18±0.11	0.31±0.03	0.20
1E96	OR	1.00	39.7	36.2±17.8	3.59±0.34	0.22±0.08	0.87	1.00	8.2	139.6±63.2	2.20±0.18	0.62±0.05	0.73
1F51	OR	1.00	7.2	37.0±24.6	3.67±0.20	0.36±0.03	0.61	1.00	2.5	218.6±144.	2.27±0.14	0.46±0.10	0.25
1FC2	OR	0.38	1.0	1837±2192	3.49±0.21	0.33±0.07	0.36	-	-	-	-	±	-
1FQJ	OR	1.00	13.9	220.5±41.2	3.15±0.44	0.42±0.06	0.42	0.78	1.2	1497±1326	2.26±0.14	0.45±0.05	0.28
1GCQ	OR	1.00	11.6	569.7±44.4	1.08±0.02	0.62±0.03	0.06	1.00	3.7	569.7±44.4	1.08±0.02	0.62±0.03	0.06
1GHQ	OR	1.00	2.2	5534±1729	3.66±0.17	0.51±0.13	1.81	-	-	-	-	±	-
1HE1	OR	1.00	7.6	908.4±201.	3.15±0.51	0.32±0.10	0.79	1.00	2.0	1179±225.5	2.00±0.06	0.50±0.05	0.68
1I4D	OR	-	-	-	-	-	-	-	-	-	-	±	-
1KAC	OR	1.00	20.6	21.0±16.5	2.50±0.08	0.59±0.03	1.10	1.00	5.0	280.9±468.	2.38±0.09	0.64±0.06	1.08
1KLU	OR	1.00	4.0	6296±1067	2.69±0.25	0.26±0.07	0.74	0.74	0.8	8729±878.2	2.21±0.12	0.33±0.03	0.59
1KTZ	OR	1.00	25.6	305.2±45.5	1.06±0.08	0.77±0.03	0.33	1.00	9.8	305.2±45.5	1.06±0.08	0.77±0.03	0.33
1KXP	OR	1.00	31.4	**19.8±8.0**	2.98±0.49	0.28±0.05	0.40	1.00	4.4	98.0±59.7	2.24±0.18	0.36±0.07	0.27
1ML0	OR	1.00	12.8	37.5±5.9	1.51±0.24	0.66±0.09	0.33	1.00	3.1	37.3±5.9	1.51±0.24	0.66±0.09	0.33
1QA9	OR	1.00	2.9	4350±905.2	3.68±0.12	0.39±0.03	0.98	-	-	-	-	±	-
1RLB	OR	1.00	18.6	662.0±244.	2.39±0.64	0.54±0.10	0.43	1.00	4.2	790.8±280	2.07±0.19	0.59±0.05	0.38
1SBB	OR	0.24	0.4	7740±1395	3.80±0.16	0.23±0.04	1.12	-	-	-	-	±	-
2BTF	OR	1.00	8.2	242.9±55.7	3.11±0.16	0.28±0.03	0.44	0.68	1.0	2215±1360	2.10±0.20	0.31±0.03	0.24
1GP2	OM	1.00	20.8	23.5±8.3	3.34±0.12	0.17±0.03	0.44	0.98	3.7	1082±1350	2.39±0.09	0.13±0.02	0.19
1GRN	OM	1.00	13.2	131.3±39.6	2.84±0.18	0.39±0.02	0.47	1.00	2.8	795.2±200.	1.91±0.14	0.32±0.03	0.20
1HE8	OM	-	-	-	-	-	-	-	-	-	-	±	-
1I2M	OM	1.00	4.3	175.4±58.2	3.55±0.18	0.26±0.05	0.67	0.16	0.2	4449±866.3	2.32±0.01	0.55±0.01	0.57
1IB1	OM	-	-	-	-	-	-	-	-	-	-	±	-
1IJK	OM	1.00	8.7	1196±146.0	1.70±0.11	0.48±0.03	0.61	1.00	2.5	1196±146	1.70±0.11	0.48±0.03	0.61
1K5D	OM	1.00	12.8	416.1±84.7	3.14±0.55	0.16±0.03	0.37	0.88	1.6	1128±1284	2.41±0.08	0.23±0.07	0.25
1M10	OM	1.00	3.1	1775±1040	3.86±0.16	0.21±0.05	0.74	-	-	-	-	±	-
1N2C	OM	1.00	2.5	2176±2309	3.72±0.21	0.23±0.04	0.34	-	-	-	-	±	-
1WQ1	OM	1.00	2.0	202.5±53.0	2.21±0.24	0.38±0.03	0.32	0.98	1.0	205.3±54.7	2.19±0.23	0.39±0.03	0.32
Avg	O	0.86	12.1	1311±437.0	2.91±0.26	0.39±0.05	0.60	0.70	2.3	1270±395.1	1.64±0.10	0.38±0.03	0.32

[a]Clasificación de casos: E:Enzima-substrato; A:Anticuerpo-Antigeno; O: Otros, R:Cuerpo Rígido (dificultad baja), M:Dificultad media.
[b]Porcentaje de las 50 ejecuciones con al menos una solución encontrada en las 10000 primeras predicciones.
[c]Número medio de soluciones encontradas en las 50 ejecuciones con $RMSD_I$<4Å (izquierda) o $RMSD_I$<2.5Å (derecha) .
[d]Posición media de la primera solución encontrada con $RMSD_I$<4Å (izquierda) o $RMSD_I$<2.5Å (derecha). Los valores de límite RMSD corresponde al criterio CAPRI descrito en [156]. Casos sombreados en gris corresponden a soluciones encontradas entre las 100 primeras posiciones. Casos en negrita corresponden a soluciones encontradas entre las 20 primeras posiciones. El símbolo "-" indica que no se ha encontrado ninguna solución entre las 10000 primeras posiciones.
[e]El valor de $RMSD_I$ ha sido calculado empleando únicamente los átomos Cα de las estructuras.
[f]Los valores de la interfaz f_{nat} and f_{not} han sido calculados siguiendo el criterio CAPRI descrito en [156].

Bibliografía

1. Chang, M.W., Lindstrom, W., Olson, A.J., and Belew, R.K.(2007) *Analysis of HIV Wild-Type and Mutant Structures via in Silico Docking against Diverse Ligand Libraries.* J. Chem. Inf. Model. **47**(3): p. 1258–1262.

2. Fox, G., Aydin, G., Gadgil, H., Pallickara, S., Pierce, M., and Wu, W. *Management of Real-Time Streaming Data Grid Services*. in *Fourth International Conference on Grid and Cooperative Computing*. 2005. Beijing, China.

3. *Griphyn*. 2000; Available from: http://www.griphyn.org/.

4. *Reaccin2*. 2007; Available from: http://www.reacciun2.edu.ve/.

5. *Portable Batch System.* 2009; Available from: http://www.openpbs.org.

6. *Load Sharing Facility*. 2009; Available from: http://www.platform.com/products/wm/LSF/.

7. *Condor*. 2009; Available from: http://www.cs.wisc.edu/condor/.

8. *Sun Grid Engine*. 2009; Available from: http://www.sun.com/gridware/.

9. *The SETI@home Project*. 2009; Available from: http://setiathome.ssl.berkeley.edu.

10. *The BOINC Project*. 2009; Available from: http://boinc.berkeley.edu/index.php.

11. Foster, I. and Kesselman, C., *The Grid: Blueprint for a New Computing Infrastructure*. 1999, San Francisco, USA: Morgan-Kaufman.

12. De Roure, D., Baker, M., and Jennings, N., *The evolution of the grid*, in *Grid Computing: Making the Global Infrastructure a Reality*, F. Berman, G. Fox, and A.J.G. Hey, Editors. 2003, John Wiley & Sons: New York, USA. p. 65-100.

13. Catlett, C. and Samarr, L.(1992) *Metacomputing.* Commun. ACM. **35**(6): p. 44-52.

14. *FAFNER*. 1995; Available from: http://www.npac.syr.edu/factoring.html.

15. Foster, I., Geisler, J., Nickless, W., Smith, W., and Tuecke, S. *Software Infrastructure for the I-WAY High Perfomance Distributed Computing Experiment*. in *Proceedings of the 5th IEEE International Symposium on High Performance Distributed Computing* 1996. Washington, DC, USA: p. 115-128.

16. Grinshaw, A. and Wulf, W.(1997) *The Legion vision of a worldwide virtual computer.* Commun. ACM. **40**(1): p. 39-45.

17. Foster, I. and Kesselman, C.(1997) *Globus: A metacomputing infrastructure toolkit.* Int. J. Supercomput. Appl. **11**(2): p. 115-128.

18. Foster, I., Kesselman, C., and Tuecke, S.(2001) *The anatomy of the Grid: Enabling Scalable Virtual Organizations.* Int. J. High Perform. Comput. Appl. **15**(3): p. 200-220.

19. Fox, G.C., *From computational science to internetics: integration of science with computer science*, in *Computational science, mathematics and software*. 2002, Purdue University Press: West Lafayette, IN, USA. p. 217-234.

20. Foster, I.(2002) *What is the Grid? - a three point checklist.* GRIDtoday. **1**(6).

21. Sotomayor, B. and Childers, L., *Globus® Toolkit 4: Programming Java Services* 2005, San Francisco, CA, USA: Morgan Kaufmann.

22. Surridge, S., Taylor, S., De Roure, D., and In, Z.E. *Experiences with GRIA; Industrial Applications on a Web Services Grid*. in *E-SCIENCE '05: Proceedings of First International Conference on e-Science and Grid Computing*. 2005. Washington, DC, USA: p. 98-105.

23. Romberg, M.(2002) *The unicore grid infrastructure.* Sci. Program. **10**(2): p. 149-157.

24. Ellert, M., Grønager, M., A., K., Kónya, B., Lindemann, J., Livenson, I., Nielsen, J.L., Niinimäki, M., Smirnova, O., and Wäänänen, A.(2007) *Advanced resource connector middleware for lightweight computational grids.* Future Gener. Comput. Syst. **23**(2): p. 219-240.

25. *Globus project*. 2009; Available from: http://www.globus.org.

26. Foster, I., Kesselman, C., Tsudik, G., and Tuecke, S. *A Security Architecture for Computational Grids*. in *Proceedings of a 5th ACM Conference on Computer and Communications Security Conference*. 1998. San Franciso, CA, USA: p. 83-92.

27. Allock, W., Bester, J., Bresnahan, J., Chervenak, A., Liming, L., Meder, S., and Tuecke, S., *GridFTP Protocol Specification.* 2002, GGF GridFTP Working Group Document.

28. Allock, W., Bresnahan, J., Foster, I., Liming, L., Link, J., and Plaszczac, P., *GridFTP Update January*. 2002, GGF GridFTP Working Group Document.

29. Bresnahan, J., Link, M., Khanna, G., Imani, Z., Kettimuthu, R., and Foster, I. *Globus GridFTP: what's new in 2007*. in *Proceedings of the first international conference on Networks for grid applications*. 2007. Lyon,France.

30. Madduri, R.K., Hood, C.S., and Allcock, W.E. *Reliable file transfer in Grid environments*. in *Proceedings of 27th Annual IEEE Conference on Local Computer Networks*. 2002. Zürich, Switzerland: p. 737-738.

31. Czajkowski, I., Foster, I., Karonis, N., Kesselman, C., Martin, S., Smith, W., and Tuecke, S. *A Resource Management Architecture for Metacomputing Systems*. in *Proceedings of the IPPS/SPDP'98 Workshop on Job Scheduling Strategies for Parallel Processing*. 1998. Orlando, FL, USA: p. 62-82.

32. *CSF. Community Scheduler Framework*. 2009; Available from: http://www.globus.org/toolkit/docs/4.0/contributions/csf/.

33. *MDS. Monitoring & Discovery System*. 2009; Available from: http://www.globus.org/toolkit/mds/.

34. *Global Grid Forum*. 2008; Available from: http://www.ggf.org.

35. Schopf, J.M., *Ten actions when superscheduling*, in *Grid Resource Management*, J. Nabrzyski, J.M. Schopf, and J. Weglarz, Editors. 2004, Kluwer Academic Publishers: Norwell, Massachussets. USA.

36. *GSB. Grid Service Broker* 2009; Available from: http://www.gridbus.org/broker/.

37. Luther, A., Buyya, R., Ranjan, R., and Venugopal, S., *Alchemi: A .NET-based Grid Computing Framework and its Integration into Global Grids*. 2003, Grid Computing and Distributed Systems Laboratory, University of Melbourne.

38. Venugopal, S., Buyya, R., and Winton, L. *A grid service broker for scheduling distributed data-oriented applications on global grids* in *MGC '04: Proceedings of the 2nd workshop on Middleware for grid computing*. 2004. New York, NY, USA: p. 75-80.

39. *GRMS. GridLab Resource Management System*. 2009; Available from: http://www.gridlab.org/WorkPackages/wp-9/index.html.

40. *GridLab Project*. 2006; Available from: http://www.gridlab.org/.

41. *EGEE user's guide: VMS Service*. 2006; Available from: https://edms.cern.ch/document/572489/1.

42. Burke, S., Campana, S., Lanciotti, E., Méndez, P., Miccio, V., Nater, C., Santinelli, R., and A., S. *GLITE 3.1 User guide Draft*. 2009; Available from: http://glite.web.cern.ch/glite/documentation/R3.2/default.asp.

43. *Enabling Grids for E-sciencE*. 2009; Available from: http://www.eu-egee.org.

44. Frey, J., Tannenbaum, T., Livny, M., Foster, I., and Tuecke, S.(2002) *Condor-G: A Computation Management Agent for Multi-Institutional Grids.* Cluster Computing. **5**(3): p. 237-246.

45. *DAGMan*. 2009; Available from: http://www.cs.wisc.edu/condor/manual/v7.0/2_10DAGMan_Applications.html.

46. Huedo, E., Montero, R.S., and Llorente, I.M.(2004) *A Framework fo Adaptative Executions in Grids.* Softw. Pract. Exper. **34**(7): p. 631-651.

47. Huedo, E., Montero, R.S., and Llorente, I.M.(2005) *The GridWay framework for adaptive scheduling and execution on grids.* Scalab. Comput. Pract. Exper. **6**(3): p. 1-5.

48. Vadhiyar, S. and Dongarra, J. *A metascheduler for the Grid.* in *Proceedings 11th IEEE International Symposium on High Performance Distributed Computing*. 2002. Edinburgh, Scotland, UK: p. 343-351.

49. *TeraGrid*. 2009; Available from: http://www.teragrid.org/.

50. *OSG. Open Science Grid.* 2009; Available from: http://www.opensciencegrid.org/, 2008.

51. *NorduGrid*. 2009; Available from: http://www.nordugrid.org/.

52. *OGF. Open Grid Forum*. 2009; Available from: http://www.gridforum.org/.

53. *DRMA. Distributed Resource Management Application API* 2009; Available from: https://forge.gridforum.org/projects/drmaa-wg.

54. *JSDL. Job Submission Description Language*. 2009; Available from: http://forge.ogf.org/sf/projects/jsdl-wg/.

55. Talbi, E. and A.Y., Z., *Grid computing for bioinformatics and computational biology*. Wiley Interscience. 2008, Hoboken, NJ, USA: John Wiley & Sons.

56. Chen, C.X. and Schmidt, B.(2005) *An adaptive grid implementation of DNA sequence alignment.* Future Gener. Comput. Syst. **21**(7): p. 988-1003.

57. Sulakhe, D., Rodriguez, A., D'Souza, M., Wilde, M., Nefedova, V., Foster, I., and Maltsev, N. *GNARE: an environment for grid based high-throughput genome analysis*. in *Fifth IEEE International Symposium on cluster computing and the Grid (CCGrid'05)*. 2005. Cardiff, Wales, UK. **1**: p. 455-462.

58. Solomonides, A., McClatchey, R., Odeh, M., Brady, M., Mulet-Parada, M., Schottlander, D., and Amendolia, S.R. *MammoGrid and eDiamond: Grids applications in mammogram analysis*. in *Proceedings of the IADIS International Conference: e-Society*. 2003. Lisbon, Portugal: p. 1032-1033.

59. Dubitzky, W., McCourt, D., Galushka, M., Romberg, M., and Schuller, B.(2004) *Grid-enabled data warehousing for molecular engineering.* Parallel Comput. **30**(9-10): p. 1019-1035.

60. Bilbao-Castro, J.R., Merinoa, A., García, I., Carazo, J.M., and Fernández, J.J.(2007) *Parameter optimization in 3D reconstruction on a large scale grid.* Parallel Comp. **33**: p. 250-263.

61. *BioinfoGRID*. 2009; Available from: http://www.bioinfogrid.eu/.

62. *OpenMolGRID*. 2005; Available from: http://www.openmolgrid.org/.

63. Stevens, R., D., Robinson, A.J., and Goble, C.A.(2003) *myGrid: Personalised bioinformatics on the information grid.* Bioinformatics. **191**(1): p. 302-304.

64. Rauwerda, H., Roos, M., Hertzberger, B., and Breit, T.(2006) *The Promise of a virtual lab.* Drug Discovery Today. **11**(5-6): p. 228-236.

65. *Genome@home*. 2009; Available from: http://genomeathome.stanford.edu/.

66. *Folding@home*. 2009; Available from: http://folding.stanford.edu/.

67. *FightAIDS@home* 2009; Available from: http://fightaidsathome.scripps.edu/.

68. Buyya, R., Branson, K., Giddy, J., and Abramson, D.(2003) *The virtual laboratory: A toolset to enable distributed molecular modelling for drug design on the world-wide grid.* Concurr. Comput. : Pract. Exper. **15**(1): p. 1-25.

69. Ewing, T.J.A. and Kuntz, I.D.(1998) *Critical evaluation of search algorithms for automated molecular docking and database screening.* Journal of Computational Chemistry. **18**(9): p. 1175-1189.

70. Jacq, N., Salzemann, J., Legré, Y., Reichstadt, M., Jacq, F., Zimmermann, M., Maaß, A., Sridhar, M., Vinod-Kusam, K., Schwichtenberg, H., Hofmann, M., and Breton, V. *WISDOM: Demonstration of in silico docking at a large scale on grid infrastructure*. in *Proceedings of Healthgrid*. 2006. Valencia, Spain.

71. Belongie, S., Malik, J., and Puzicha, J.(2002) *Shape matching and object recognition using shape contexts.* IEEE Trans. Pattern Anal. Match. Intell. **24**(4): p. 509-522.

72. Sebastian, T., Klein, P., and Kimia, B. *Shock-Based Indexing into Large Shape Databases*. in *ECCV 2002.* 2002. Copenhagen, Denmark: p. 731-746.

73. Yang, X., Bai, X., Yu, D., and Latecki, L.J.(2008) *Shape Classification Based on Skeleton Path Similarity.* IJPRAI. **22**(4): p. 733-746.

74. Tangelder, J.W. and Veltkamp, R.C.(2008) *A survey of content based 3D shape retrieval methods.* Multimedia Tools Appl. **39**(3): p. 441-471.

75. Zhang, C. and Chen, T. *Indexing and retrieval of 3D models aided by active learning*. in *MULTIMEDIA '01: Proceedings of the ninth ACM international conference on Multimedia*. 2001. Otawa, Canada: p. 615-616.

76. Rea, H.J., Sung, R., and J.R., C. *Identifying 3D object features using shape distributions*. in *Proc. 18th international conference on computer aided production engineering*. 2003. Edinburg, Scotland, U.K.

77. Yu, M., Atmosukarto, I., Leow, W.K., Huang, Z., and Xu, R. *3D model retrieval with morphing-based geometric and topological feature maps*. in *IEEE conf. on computer vision and pattern recognition*. 2003. Madison, WI, USA: p. 656-661.

78. Biasotti, S., Marini, S., Mortara, M., and Patané, G. *An overview of properties and efficacy of topological skeletons in shape modelling*. in *Proc. shape modeling international*. 2003. Seoul, Korea: p. 245-254.

79. Riesenhuber, M. and Poggio, T. *Computational models of object recognition in cortex: a review*. in *Technical report CBCL no. 1695*. 2000.

80. Volkmann, N. and Hanein, D.(2003) *Docking of atomic models into reconstructions from electron microscopy.* Methods Enzymol. **374**: p. 204-225.

81. Ritchie, D.W.(2008) *Recent progress and future directions in protein-protein docking.* Current Protein and Peptide Science. **9**(1): p. 1-15.

82. Ankerst, M., Kastenmuller, G., Kriegel, H.-P., and Seidl, T. *3D Shape histograms for similarity search and classification in spatial databases*. in *Proc. symposium in large databases*. 1999. Hong Kong, China: p. 207-266.

83. Katchalski-Katzir, E., Shariv, I., Eisenstein, M., Friesem, A.A., Aflalo, C., and Vakser, I.A.(1992) *Molecular surface recognition: determination of geometric fit between proteins and their ligands by correlation techniques.* Proc. Natl. Acad. Sci. USA. **89**(6): p. 2195-2199.

84. Wriggers, W. and Chacon, P.(2001) *Modeling tricks and fitting techniques for multiresolution structures.* Structure (Camb). **9**(9): p. 779-788.

85. Eisenstein, M. and Katchalski-Katzir, E.(2004) *On proteins, grids, correlations, and docking.* C. R. Biologies. **327**(5): p. 409-420.

86. Ritchie, D.W. and Kemp, G.J.(2000) *Protein docking using spherical polar Fourier correlations.* Proteins. **39**(2): p. 178-194.

87. Kovacs, J.A. and Wriggers, W.(2002) *Fast rotational matching.* Acta Crystallogr. D. Biol. Crystallogr. **58**(8): p. 1282-1286.

88. Dauter, Z.(2006) *Current state and prospects of macromolecular crystallography.* Acta Crystallogr. D. Biol. Crystallogr. **62**(1): p. 1-11.

89. Wüthrich, K.(1998) *The second decade — into the third millenium.* Nat. Struc. Biol. **5 Suppl**: p. 492-495.

90. *PDB. Protein Data Bank*. 2009; Available from: http://www.rcsb.org.

91. Frank, J.(2002) *Single-particle imaging of macromolecules by cryo-electron microscopy.* Annu. Rev. Biophys. Biomol. Struct. **31**(1): p. 303-319.

92. Wriggers, W., Milligan, R.A., and McCammon, J.A.(1999) *Situs: A package for docking crystal structures into low-resolution maps from electron microscopy.* J. Struct. Biol. **125**(2-3): p. 185-195.

93. Volkmann, N. and Hanein, D.(1999) *Quantitative Fitting of Atomic Models into Observed Densities Derived by Electron Microscopy.* J. Struct. Biol. **125**(2-3): p. 176-184.

94. Roseman, A.M.(2000) *Docking structures of domains into maps from cryo-electron microscopy using local correlation.* Acta Crystallogr. D. Biol. Crystallogr. **56**(10): p. 1332-1340.

95. Rossmann, M.G.(2000) *Fitting atomic models into electron-microscopy maps.* Acta Crystallogr. D. Biol. Crystallogr. **56**(10): p. 1341-1349.

96. Halperin, I., Ma, B., Wolfson, H., and Nussinov, R.(2002) *Principles of docking: An overview of search algorithms and a guide to scoring functions.* Proteins. **47**(4): p. 409-443.

97. Burgoyne, N.J. and Jackson, R.M.(2006) *Predicting protein interaction sites: binding hot-spots in protein-protein and protein-ligand interfaces.* Bioinformatics. **22**(11): p. 1335-1342.

98. Halperin, I., Wolfson, H., and Nussinov, R.(2004) *Protein-protein interactions: coupling of structurally conserved residues and of hot spots across interfaces. Impications for docking.* Structure. **12**(6): p. 1027-1038.

99. Fernandez-Recio, J., Totrov, M., Skorodumov, C., and Abagyan, R.(2005) *Optimal docking area: a new method for predicting protein-protein interaction sites.* Proteins. **58**(1): p. 134-143.

100. Yao, H., Kristensen, D.M., Mihalek, I., Sowa, M.E., Shaw, C., Kimmel, M., Kavraki, L., and O., L.(2003) *An accurate, sensitive, and scalable method to identify functional sites in protein structures.* J. Mol. Biol. **326**(1): p. 255-261.

101. Henrick, K. and Thornton, J.M.(1998) *PQS: a protein quaternary structure file server.* Trends Biochem. Sci. **23**(9): p. 358-361.

102. Preiβner, R., Goede, A., and Frömmel, C.(1998) *Dictionary of interfaces in proteins (DIP). Data bank of complementary molecular surface patches.* J. Mol. Biol. **280**(3): p. 535-550.

103. Jefferson, E.R., Walsh, T.P., and Barton, G.J.(2006) *Biological units and their effect upon the properties and prediction of protein-protein interactions.* J. Mol. Biol. **364**(5): p. 1118-1129.

104. Aloy, P. and Russell, R.B.(2003) *InterPreTS: Protein interaction prediction through tertiary structure.* Bioinformatics. **19**(1): p. 161-162.

105. Rajamani, D., Thiel, S., Vajda, S., and Camacho, C.J.(2004) *Anchor residues in protein-protein interactions.* Proc. Natl. Acad. Sci. USA. **101**(31): p. 11287-11292.

106. Zacharias, M.(2003) *Protein-protein docking with a reduced protein model accounting for side-chain flexibility.* Prot. Sci. **12**(6): p. 1271-1282.

107. Bastard, K., Prévost, C., and Zacharias, M.(2006) *Accounting for loop flexibility during protein-protein docking.* Proteins. **62**(4): p. 956-969.

108. Cavasotto, C.N., Kovacs, J.A., and Abagyan, R.(2005) *Representing receptor flexibility in ligand docking through relevant normal modes.* J. Am. Chem. Soc. **127**(26): p. 9632-9640.

109. Mustard, D. and Ritchie, D.W.(2005) *Docking essential dynamics eigenstructures.* Proteins. **60**(2): p. 269-274.

110. Chen, R., Li, L., and Weng, Z.(2003) *ZDOCK: an initial-stage protein-docking algorithm.* Proteins. **52**(1): p. 80-87.

111. Kozakov, D., Brenke, R., Comeau, S.R., and Vajda, S.(2006) *PIPER: an FFT-based protein docking program with pairwise potentials.* Proteins. **65**(2): p. 392-406.

112. Barish, G. and Obraczka, K.(2000) *World Wide Web caching: Trends and techniques.* IEEE Communication Magazine. **38**(5): p. 178-184.

113. Bester, J., Foster, I., Kesselman, C., Tedesco, J., and Tuecke, S. *GASS: A Data Movement and Access Service for Wide Area Computing Systems.* in *6th Workshop on I/O in Parallel and Distributed Systems*. 1999. Atlanta, Georgia, EEUU.

114. Yonny, C., Pierson, J.M., and Brunie, L.(2007) *Management of a Cooperative Cache in Grids with Grid Cache Services.* Concurr. Comput. : Pract. Exper. **19**(16): p. 2141-2155.

115. Tierney, B., Johnston, W., and Lee, J. *A Cache-Based Data Intensive Distributed Computing Architecture For "grid" Applications*. in *The CERN School of Computing*. 2000. Marathon, Greece.

116. Shoshani, A., Sim, A., and Gu, J., *Storage resource managers: essential components for the Grid*, in *Grid resource management: state of the art and future trends*, J. Nabrzyski, J.M. Schopf, and J. Węglarz, Editors. 2004, Kluwer Academic Publishers: Norwell, MA, USA. p. 321 - 340.

117. Liu, F. and Song, F. *An Optimization of Resource Replication Access in Grid Cache*. in *Advances in Grid and Pervasive Computing: First International Conference*. 2008. Taichung, Taiwan: p. 83-92.

118. Chervenak, A., Schuler, R., Kesselman, C., Korada, S., and Moe, B. *Wide Area Data Replication for Scientific Collaborations*. in *6th IEEE/ACM International Workshop on Grid Computing* 2005. Washington DC, USA

119. Garzon, J.I., Huedo, E., Montero, R.S., Llorente, I.M., and Chacon, P.(2009) *End-to-end Cache System for Grid Computing: Design and Efficiency Analysis of a High-throughput Bioinformatic Docking Application.* IJHPCA. **OnlineFirst**.

120. Rivest, R. *The MD5 Message-Digest Algorithm (RFC 1321)*. in *www.ietf.org*. 1992.

121. Wang, J.(1999) *A survey of web caching schemes for the internet.* ACM Comput. Commun. Rev. **29**(5): p. 36-46.

122. Crowter, R.A., in *The Molecular Replacement Method*, M.G. Rosmann, Editor. 1972, Gordon & Breach: London, U.K. p. 173-178.

123. Hobson, E.W., *The Theory of Spherical and Ellipsoidal Harmonics*. 1931, Cambridge, U.K.: Cambridge University Press.

124. Goldstein, H., *The Euler Angles and Euler Angles in Alternate Conventions*, in *Classical Mechanics*, M. Reading, Editor. 1980, Addison-Wesley: Boston, MA, USA. p. 143-148,606-610.

125. McWeeny, R. Symmetry: An Introduction to Group Theory and its Applications 2002, Mineola, NY, USA: Dover. 171-174.

126. Brink, D.M. and Satchler, G.R., *Angular Momentum*. 1993, Oxford, U.K.: Claredon Press.

127. Risbo, T.(1996) *Fourier Transform summation of Legendre series and D-functions.* J. Geodes. **70**(7): p. 383-396.

128. Healy, D., Rockmore, D., Kostelec, P., and Moore, S.(2003) *FFTs for the 2-Sphere - Improvements and Variations.* The Journal of Fourier Analysis and Applications. **9**(4): p. 341-385.

129. Driscoll, J.R. and Healy, D.(1994) *Computing Fourier transforms and convolutions on the 2-sphere.* Adv. in Appl. Math. **15**(2): p. 202-250.

130. Kovacs, J.A., Chacon, P., Cong, Y., Metwally, E., and Wriggers, W.(2003) *Fast rotational matching of rigid bodies by fast Fourier transform acceleration of five degrees of freedom.* Acta Crystallogr. D. Biol. Crystallogr. **59**(8): p. 1371-1376.

131. Fabiola, F. and Chapman, M.S.(2005) *Fitting of high-resolution structures into electron microscopy reconstruction images.* Structure (Camb). **13**(3): p. 389-400.

132. Fiser, A. and Sali, A.(2003) *Modeller: generation and refinement of homology-based protein structure models.* Methods Enzymol. **374**: p. 461-491.

133. Garzon, J.I., Kovacs, J., Abagyan, R., and Chacon, P.(2007) *ADP_EM: fast exhaustive multi-resolution docking for high-throughput coverage.* Bioinformatics. **23**(4): p. 427-433.

134. Dill, K.A. and Bromberg, S., *Molecular Driving Forces: Statistical Thermodynamics in CHemistry & Biology*. 2003, London, U.K.: Taylor & Francis.

135. Trotov, M., *Potein-Ligand Docking as an Energy Optimization Problem*, in *Drug-Receptor Thermodynamics: Introduction and Application*, R.B. Raffa, Editor. 2001, Wiley & Sons: Hoboken, NJ, USA.

136. Gabb, H.A., Jackson, R.M., and Sternberg, M.J.(1997) *Modelling protein docking using shape complementarity, electrostatics and biochemical information.* J. Mol. Biol. **272**(1): p. 106-120.

137. Heifetz, A., Katchalski-Katzir, E., and Eisenstein, M.(2002) *Electrostatics in protein-protein docking.* Protein Sci. **11**(3): p. 571-587.

138. Moont, G., Gabb, H.A., and Sternberg, M.J.(1999) *Use of pair potentials across protein interfaces in screening predicted docked complexes.* Proteins. **35**(3): p. 364-373.

139. Fernandez-Recio, J., Totrov, M., and Abagyan, R.(2002) *Soft protein-protein docking in internal coordinates.* Protein Sci. **11**(2): p. 280-291.

140. Abagyan, R.A., *Protein structure prediction by global energy optimization. Computer Simulation of Biomolecular Systems: Theoretical and Experimental Applications*, ed. W.F. van Gunsteren, P.K. Weiner, and A.J. Wilkinson. Vol. 3. 1997, Leiden, The Netherlands: ESCOM Science Publishers BV.

141. Fernandez-Recio, J., Totrov, M., and Abagyan, R.(2004) *Identification of protein-protein interaction sites from docking energy landscapes.* J. Mol. Biol. **335**(3): p. 843-865.

142. Garzon, J.I., Lopez-Blanco, J.R., Pons, C., Kovacs, J., Abagyan, R., Fernandez-Recio, J., and Chacon, P.(2009) *FRODOCK: a new approach for fast rotational protein-protein docking.* Bioinformatics. **25**(19): p. 2544-2551.

143. Garzon, J.I., Huedo, E., Montero, R.S., Llorente, I.M., and Chacon, P.(2007) *Adaptation of a Multi-Resolution Docking Bioinformatics Application to the Grid.* J. Softw. **2**(2): p. 1-10.

144. Ceulemans, H. and Russell, R.B.(2004) *Fast fitting of atomic structures to low-resolution electron density maps by surface overlap maximization.* J. Mol. Biol. **338**(4): p. 783-793.

145. Chacon, P. and Wriggers, W.(2002) *Multi-resolution contour-based fitting of macromolecular structures.* J. Mol. Biol. **317**(3): p. 375-384.

146. Martin-Benito, J., Boskovic, J., Gomez-Puertas, P., Carrascosa, J.L., Simons, C.T., Lewis, S.A., Bartolini, F., Cowan, N.J., and Valpuesta, J.M.(2002) *Structure of eukaryotic prefoldin and of its complexes with unfolded actin and the cytosolic chaperonin CCT.* Embo J. **21**(23): p. 6377-6386.

147. Craighead, J.L., Chang, W.H., and Asturias, F.J.(2002) *Structure of yeast RNA polymerase II in solution: implications for enzyme regulation and interaction with promoter DNA.* Structure (Camb). **10**(8): p. 1117-1125.

148. Asturias, F.J.(2004) *RNA polymerase II structure, and organization of the preinitiation complex.* Curr. Opin. Struct. Biol. **14**(2): p. 121-129.

149. Xiang, Z.(2006) *Advances in homology protein structure modeling.* Curr. Protein Pept. Sci. **7**(3): p. 217-227.

150. Topf, M. and Sali, A.(2005) *Combining electron microscopy and comparative protein structure modeling.* Curr. Opin. Struct. Biol. **15**(5): p. 578-585.

151. *Salilab. Laboratory of Andrej Sali*. 2007; Available from: http://salilab.org.

152. Topf, M., Baker, M.L., John, B., Chiu, W., and Sali, A.(2005) *Structural characterization of components of protein assemblies by comparative modeling and electron cryo-microscopy.* J. Struct. Biol. **149**(2): p. 191-203.

153. Ortiz, A.R., Strauss, C.E., and Olmea, O.(2002) *MAMMOTH (matching molecular models obtained from theory): an automated method for model comparison.* Protein Sci. **11**(11): p. 260626-21.

154. Jiang, W., Baker, M.L., Ludtke, S.J., and Chiu, W.(2001) *Bridging the information gap: computational tools for intermediate resolution structure interpretation.* J. Mol. Biol. **308**(5): p. 1033-1044.

155. Mintseris, J., Wiehe, K., Pierce, B., Anderson, R., Chen, R., Janin, J., and Weng, Z.(2005) *Protein-Protein Docking Benchmark 2.0: an update.* Proteins. **60**(2): p. 214-216.

156. Mendez, R., Leplae, R., De Maria, L., and Wodak, S.J.(2003) *Assessment of blind predictions of protein-protein interactions: current status of docking methods.* Proteins. **52**(1): p. 51-67.

157. Chen, R., Li, L., and Weng, Z.(2003) *ZDOCK: an initial-stage protein-docking algorithm.* Proteins. **52**(1): p. 80-87.

158. Janin, J., Henrick, K., Moult, J., Eyck, L.T., Sternberg, M.J., Vajda, S., Vakser, I., and Wodak, S.J.(2003) *CAPRI: a Critical Assessment of PRedicted Interactions.* Proteins. **52**(1): p. 2-9.

159. Ritchie, D.W., Kozakov, D., and Vajda, S.(2008) *Accelerating and Focusing Protein-Protein Docking Correlations Using Multi-Dimensional Rotational FFT Generating Functions.* Bioinformatics. **24**(17): p. 1865-1873.

160. Montero, R.S., Huedo, E., and Llorente, I.M.(2006) *Benchmarking of High Throughput Computing Applications on Grids.* Parall. Comp. **32**(4): p. 267-279.

161. Hockney, R.W. and Jesshope, C.R., *Parallel Computers 2: Architecture and Algorithms*. 1988, Bristol, U.K.: Adam Hilger Ltd.

162. Frumkin, M. and Van der Wijngaart, R.F.(2002) *NAS Grid Benchmarks: a tool for Grid space exploration.* Cluster Computing. **5**(3): p. 247 - 255.

163. Bailey, D.H., Barszcz, E., and Barton, J.T.(1991) *The NAS Parallel Benchmarks.* Int. J. Supercomp. App. **5**(3): p. 63-73.

164. Bailey, D.H., Harris, T., Van der Wigngaart, R., Saphir, W., Woo, A., and Yallow, M., *The NAS Parallel Benchmarks 2.0*, in *Technical Report NAS-95-010 NASA Ames Research Center*. 1995.